Институт проблем информационной
безопасности Московского государственного
университета имени М. В. Ломоносова

Академия криптографии Российской Федерации

Математика и безопасность информационных технологий

Материалы конференции в МГУ
28–29 октября 2004 г.

Москва
Издательство МЦНМО
2005

ББК 32.81В6
М34

Математика и безопасность информационных технологий. М34 Материалы конференции в МГУ 28–29 октября 2004 г. — М.: МЦНМО, 2005. — 384 с.

Спонсоры конференции:
ЗАО «Софтлайн-М»
Ассоциация Защиты Информации
ЗАО «ДиалогНаука»
ФГУП «Пензенский филиал НТЦ “Атлас”»
ФГУП «НИИ “Квант”»
ООО «Крипто-Про»
Компания «РНТ»
Компания «Систематика»
ЗАО «Санкт-Петербургский РЦЗИ»

МАТЕМАТИКА И БЕЗОПАСНОСТЬ
ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Материалы конференции в МГУ 28–29 октября 2004 г.

Издательство Московского центра непрерывного математического образования. 119002, Москва, Большой Власьевский пер., 11.

Лицензия ИД № 01335 от 24.03.2000 г. Подписано к печати 15.09.2005 г. Формат 60 × 90/16. Печать офсетная. Объем 24 печ. л. Тираж 500 экз. Заказ № .

Отпечатано с готовых диапозитивов в ГУП «Облиздат». 248640, г. Калуга, пл. Старый торг, 5.

ISBN 5-94057-212-X

© Коллектив авторов, 2005.
© МЦНМО, 2005.

Оглавление

Приветственные выступления

Пленарные доклады

Секция «Математические проблемы информационной безопасности»

Секция «Информационная безопасность компьютерных систем»

Программа конференции

Четверг, 28 октября 2004 г.
Пленарные заседания

Приветственное выступление помощника Секретаря Совета безопасности РФ, директора ИПИБ МГУ В. П. Шерстюка.

Приветственное выступление вице-канцлера Университета штата Нью-Йорк (SUNY), д-ра Елизаветы Д. Капальди (Dr. Elizabeth D. Capaldi).

Приветственное выступление первого заместителя председателя Комитета по местному самоуправлению Государственной Думы РФ, д-ра физ.-мат. наук, профессора Г. К. Сафаралиева.

Пленарные доклады:

Б. А. Погорелов, *В. Н. Сачков*, *В. В. Ященко*. О терминологической базе современной криптографии.

М. М. Глухов, *Б. А. Погорелов*. О некоторых применениях групп в криптографии.

Перерыв.

В. А. Васенин, *А. В. Галатенко*. Математические модели распределенных компьютерных систем.

Н. П. Варновский, *Е. А. Голубев*, *О. А. Логачёв*. Современные направления стеганографии.

Обед.

В. А. Васенин, *А. В. Галатенко*, *В. В. Корнеев*, *А. А. Макаров*. Математическое и программное обеспечение активного аудита больших распределенных систем.

Н. П. Варновский, *В. А. Захаров*, *Н. Н. Кузюрин*. Математические проблемы обфускации.

Scott D. Stoller, *Yanhong A. Liu*. Security policy languages and enforcement.

Tzi-cker Chiueh. Program semantic-aware intrusion detection.

Пятница, 29 октября 2004 г.
Секционные заседания

Секция «Математические проблемы информационной безопасности»

Руководитель секции — О. А. Логачёв, зав. отделом ИПИБ МГУ.

Н. П. Варновский, *Е. А. Голубев*, *О. А. Логачёв*. Статистическое описание звукового стегоконтейнера.

В. С. Анашин. О поточных шифраторах с динамически изменяющимся законом шифрования.

М. А. Пудовкина. Об алгоритме поточного шифрования VR.

Е. В. Кутырева. О числе ненулевых элементов треугольника Паскаля над конечным полем.

Ю. В. Таранников. О значениях аффинного ранга носителя спектра платовидных функций.

Перерыв.

А. В. Покровский, *М. С. Никифоров*. О свойствах аффинных и квадратичных аннуляторов булевых функций.

Ю. С. Харин, *А. Н. Ярмола*. Дискретные временные ряды с «длинной памятью» и их использование в задачах защиты информации.

А. А. Ботев. Об алгебраической иммунности рекурсивных конструкций нелинейных фильтров.

Б. Я. Рябко, *В. А. Монарев*, *Ю. И. Шокин*. Новый класс статистических тестов для случайных чисел и его применение в задачах криптографии.

Ю. В. Виланский. Криптографический примитив MV2.

К. Н. Панков. Асимптотическая формула для числа (n, m, k)-устойчивых функций.

О. А. Козлитин. Теоретико-групповая характеризация неавтономных линейных регистров сдвига над свободным модулем.

Обед.

О. А. Логачёв, *А. А. Сальников*, *В. В. Ященко*. Многогранники в конечной абелевой группе и их криптографические приложения.

В. М. Фомичёв. О распознавании признаков элементов в конечных группах.

А. Ю. Нестеренко. YAQS — еще один алгоритм квадратичного решета.

М. А. Черепнёв. О величине простых делителей чисел вида $p - 1$.

А. В. Саранцев. Регулярные системы однотипных двоичных функций.

А. Г. Ростовцев. Защита от side channel attack на основе случайных изоморфизмов.

Н. В. Никонов. О классификации обобщенных запретов всех булевых функций от 3-х переменных.

Стендовые доклады:

А. В. Бабаш. О нахождении минимальных систем областей импримитивности.

А. В. Бабаш. Сильно приведенные автоматы.

П. А. Беляков, *В. Г. Никонов*. Синтез латинских квадратов с помощью полиномов двух переменных над кольцом $\mathbb{Z}_{2^k}$.

М. А. Буряков, *О. А. Логачёв*. О распределении уровня аффинности на множестве булевых функций.

Д. А. Куропаткин. О фрагментах наборов чисел, полученных приведением по модулю два перестановок с ограниченным числом инверсий.

А. В. Лапшин. Задачи нахождения оценок параметров распределения слагаемого по наблюдениям суммы случайных величин на конечной абелевой группе.

В. А. Мищенко. Многоканальные криптографические системы и их применение.

В. М. Сидельников. О построении дизъюнктных (superimposed) кодов и их использовании в криптографии.

Секция «Информационная безопасность компьютерных систем»

Руководитель секции — В. А. Васенин, зав. отделом ИПИБ МГУ.

А. А. Грушо, *Е. Е. Тимонина*. Враждебные многоагентные системы.

И. В. Котенко. Многоагентные модели противоборства злоумышленников и системы защиты в сети Интернет.

А. А. Грушо, *Е. Е. Тимонина*. Математическая модель безопасного взаимодействия.

А. В. Тишков, *И. В. Котенко*. Исчисление событий для спецификации и верификации политик безопасности защищенной вычислительной сети.

E. Rich. An emerging perspective on the dynamics of insider cyber threats.

Перерыв.

В. Б. Бетелин, *С. Г. Бобков*, *В. А. Галатенко*, *А. Н. Годунов*, *А. И. Грюнталь*, *А. Г. Кушниренко*, *П. Н. Осипенко*. Анализ тенденций развития аппаратно-программного обеспечения и их влияние на информационную безопасность.

В. С. Заборовский, *Ю. А. Шеманин*. Архитектура распределенных сетевых процессоров: особенности применения в системах информационной безопасности.

О. В. Казарин. Три гипотезы в защите программ.

H. G. Berg. A teaching Hospital Approach to identifying risks, research problems and teaching materials in information security.

Обед.

Д. П. Зегжда, *А. М. Вовк*. Защищенная гибридная операционная система «Linux over Феникс».

В. К. Попков, *О. Д. Соколова*, *А. Н. Юргенсон*. О некоторых задачах анализа устойчивости работы информационных сетей.

А. С. Родионов, *О. К. Родионова*, *Д. А. Мигов*, *М. Ю. Мурзин*. Использование метода ветвления для точного расчета вероятности связности случайного графа.

В. В. Платонов. Архитектура системы предотвращения вторжений.

Перерыв.

Ф. М. Пучков, *К. А. Шапченко*. К вопросу о выявлении возможных переполнений буферов посредством статического анализа исходного кода программ.

С. С. Корт. Операционный анализ безопасности поведения программного обеспечения.

О. О. Андреев. Сравнение ролевой и дискреционной моделей разграничения доступа.

С. А. Ахманов, *Д. А. Надежкин*, *Д. А. Раевский*. Дополнительные механизмы обеспечения безопасности в ядре ОС Linux: фильтр системных вызовов, протоколирование системных вызовов, квотирование ресурсов на уровне пользователя.

Проблемы обеспечения информационной безопасности в современном мире

В. П. Шерстюк

Уважаемые участники конференции!

Рад поздравить Вас с началом работы Третьей общероссийской конференции «Математика и безопасность информационных технологий». Время показало актуальность и востребованность у научной и производственной общественности, а также представителей государственных органов исполнительной власти и бизнеса тематики настоящей конференции. Она заняла самостоятельную нишу в перечне конференций, посвященных проблеме обеспечения информационной безопасности и ее авторитет неуклонно растет. Об этом свидетельствует не только возросшее представительство российских участников конференции, но и проявленный интерес к ней наших зарубежных коллег. Пользуясь возможностью, рад поприветствовать в стенах первого вуза России наших уважаемых гостей, вице-канцлера Университета штата Нью-Йорк, д-ра Элизабет Д. Капальди и ее коллег, а также представителей Белоруссии, Швейцарии и Украины.

Актуальность рассматриваемых на конференции проблем обусловлена прежде всего тем, что в современном мире информация стала стратегическим национальным ресурсом, одним из основных богатств государства, претендующего на достойное место в международном сообществе. Сегодня экономический, оборонный и политический потенциал страны в значительной мере определяется уровнем развития ее информационной инфраструктуры, но при этом пропорционально возрастает и уязвимость национальных информационных ресурсов по отношению к негативным воздействиям. Вышесказанное в полной мере относится и к России, так как ее стремительная информатизация, вызванная проходящими политическими и экономическими процессами, ведет к тому, что благополучие страны все в большей степени зависит от безопасности ее информационных ресурсов.

В условиях вступления России в мировое информационное пространство особую значимость приобретают задачи *защиты нацио-*

нальных интересов страны в глобальной информационно-телекоммуникационной среде.

Стремительное развитие и внедрение информационных технологий оказывает возрастающее влияние на все стороны жизнедеятельности государства и общества. Социальная, политическая, экономическая и военная сферы находятся в прямой зависимости от работы вычислительных и информационных сетей, систем связи и управления, различных электронных средств, программного обеспечения, составляющих техническую базу информационного пространства и одновременно определяющих его уязвимость.

Менее чем за одно поколение информационная революция и внедрение компьютеров фактически во все сферы нашего общества изменили характер работы экономики, порядок и принципы обеспечения нами нашей безопасности и характер построения нашей повседневной жизни. Очевидно, что в ближайшем будущем роль информационных технологий будет всемерно усиливаться, а сращивание систем, обеспечивающих национальную безопасность, с коммерческими и широкодоступными системами и сетями будет еще большей.

С другой стороны новое многообещающее время таит в себе новые опасности, связанные с развитием информационных технологий. Современные тенденции развития общества ведут к разрастанию международного терроризма, появлению на его вооружении новейших технологий и возникновению принципиально нового высокотехнологичного кибертерроризма. При этом могут применяться как информационно-компьютерные, так и информационно-психологические средства. Так, сеть Интернет активно используется для распространения идеологии терроризма, вовлечения в противоправную деятельность новых членов, о чем красноречиво свидетельствует наличие большого количества соответствующих сайтов. Такие тенденции, как глобализация экономики, информатизация всех жизненно важных сфер деятельности общества, популяризация информационных технологий дают основание говорить о большой вероятности акций компьютерного терроризма в ближайшем будущем.

Мы знаем, что эта угроза реальна. В условиях формирования единого информационного пространства террористы могут для осуществления своих замыслов превратить любую ПЭВМ в мощное оружие, способное нанести огромный, порой трудно восполнимый ущерб. Все автоматические системы чувствительны к несанкционированному доступу и уничтожению информации. Согласованная попытка нарушения защиты компьютеров в любом из наших ключевых экономических секторов или федеральных и региональных органов власти могла бы

иметь катастрофические инфраструктурные последствия. Все это ставит ряд принципиально новых проблем в обеспечении национальной безопасности.

Поэтому защитные меры в данной области приобретают первостепенное значение не только для обеспечения работы военных систем, но также правительственных и гражданских структур, поддерживающих нормальное функционирование всех компонентов жизнедеятельности государства. Если мы хотим в полной мере пользоваться преимуществами постиндустриального «информационного века», сохранить нашу безопасность и защитить наше экономическое благополучие, мы должны защитить наши важнейшие автоматизированные системы от несанкционированного разрушающего воздействия.

Проявившиеся в последнее время факторы системной методологии применения новых информационных технологий в государственных и коммерческих ИТКС решительно меняют подходы к обеспечению безопасности шифр-систем, определению критериев стойкости шифрования, существенно влияют на формирование списка научных проблем, нуждающихся в приоритетной проработке. К наиболее актуальным факторам, определяющим проблематику современных реалий в области создания средств защиты информации в ИТКС, на наш взгляд можно отнести следующие.

1. Развитие систем связи и фактические превращения их в компьютерные системы связи, по новому ставит задачу защиты информации и указывают на новое положение принципов шифрования в информационных системах. Если раньше шифрование применялось только для защиты информации передаваемой по каналу связи, то теперь без шифрования и использования криптографической электронно-цифровой подписи невозможно построить эффективную защиту от НСД как самой информации в ИТКС, так и систем управления и реализации связных функций. *Фактически понятие криптографии расширилось и, наряду с созданием стойких алгоритмов засекречивания информации, представляет собой методологию и технологию обеспечения безопасности компьютерных систем.*

2. *Шифрсредства реализуются в открытых средах*, а именно, на аппаратно-программных платформах вычислительной техники; *шифрсредства выносятся на рабочие места, интегрируются с абонентским и сетевым оборудованием*; получают распространение *шифрсредства персонального пользования*, такие как переносные рабочие места на ноутбуках, аппаратура сотовой связи, пейджеры, интеллектуальные пластиковые карты различного целевого назначения и т. д.

Эксплуатация шифрсредств вне шифрорганов по-новому ставит проблему разграничения доступа к шифрсредствам. Если до этого обоснованно считалось, что данная проблема надежно решалась организационными мерами, то теперь с учетом описанных выше процессов необходимо коренным образом менять идеологию защиты. *В указанных выше новых условиях эксплуатации шифрсредств их стойкость начинает значительным образом определяться стойкостью физических мер защиты криптографически опасной информации*.

3. Становятся широко доступными сложнейшие методы анализа электронных устройств — как их внутренней структуры, так и внешних проявлений внутренних процессов, происходящих при выполнении ими своих функций. Это делает еще более *актуальными разработку недорогих и эффективных средств физической защиты криптографических модулей*, *носителей ключей*, *паролей и другой критической информации*.

4. Наблюдается *резкое изменение аппаратной платформы из-за необходимости обеспечения большей производительности*, *резкого увеличения пропускной способности каналов связи*. Наблюдается массовое внедрение *волоконно-оптических линий* связи начиная с абонентского и до магистрального уровней. Это оказывает большое влияние на *создание новой аппаратной платформы с использованием оптических методов обработки информации*. Получают развитие *квантово-механические средства обработки информации и методы*, *основанные на идеях «детерминированного хаоса»*.

5. Все более широкое использование новых физических эффектов для построения средств обработки, передачи и защиты информации делает криптографию и все, что связано с защитой информации, более чувствительной к кризисным явлениям, наблюдаемым в современной физике и физической парадигме мира, что усугубляется резким свертыванием у нас в стране соответствующих фундаментальных исследований.

6. По мере того, как национальные, региональные и глобальные информационные инфраструктуры приобретают все большее значение в жизни государств образованное ими *киберпространство становится ареной все более опасного информационного терроризма и информационного межгосударственного противоборства*.

Перечисленные проблемные области требуют комплексного научного обеспечения

- криптографического,
- правового — внутри страны, в рамках СНГ, на международном уровне,
- общенаучного, и, прежде всего — в области математики, физики и информатики.

Безусловно, решение такого комплекса задач требует консолидации усилий всего научно-производственного потенциала страны.

Кстати говоря, Дэвид Канн, известный американский историк криптографии дал такое определение великой державе: «*Великая держава — это страна*, *которая обладает ядерными технологиями*, *ракетными технологиями и криптографией*».

Современная криптография базируется на последних достижениях математики, ряда фундаментальных физических и инженерных дисциплин и новейших достижениях в области компьютерных технологий.

Необходимость обоснования специальных свойств перспективных криптосхем, а также ряд новых задач, постоянно возникающих в теоретической криптографии в течение последнего десятилетия (разработка методов аутентификации информации и проблема создания надежной электронно-цифровой подписи, построение ключевых систем, устойчивых к компрометации части абонентов сети, разработка криптографических протоколов, исследования и практическое использование криптосистем с открытым ключом), обуславливает *необходимость непрерывного развития математических методов криптографических исследований*. В настоящее время решить эти проблемы можно только на основе широкого привлечения специалистов из смежных областей науки и техники.

Но качество систем защиты определяется не только реализованными в них математическими алгоритмами. Для построения надежной защищенной информационной системы необходимо решить *комплекс инженерно-физических проблем*. *Принципиально новым направлением защиты наиболее важных компонентов компьютерных систем в настоящее время является построение физико-технической защиты таких важнейших элементов как пароли*, *ключи*, *криптосхемы и др*.

Современные электронные программные средства криптозащиты (шифрования, электронно-цифровой подписи, защиты от несанкционированного доступа) являются сложными системами взаимодействующих дискретных автоматов и оценка их криптографических параметров является трудной научно-технической проблемой, не имеющей в мировой практике каких-либо стандартных решений.

Криптографическая наука представляет собой особое научное направление, основанное не только на математических методах и подхо-

дах к исследованию сложности алгоритмов шифрования и оптимизации (синтезе) последних, но и на элементах физики и радиотехники. Поэтому только на основе интеграции указанных подходов возможно достижение положительных результатов при разработке принципиально новых методов защиты информации.

Традиционные направления фундаментальных исследований в области теоретической и инженерной криптографии всегда были связаны не только с активным использованием современных достижений математики, физики, радиоэлектроники, информатики, но и с их существенным развитием и получением новых научных результатов, вносящих вклад в отечественную криптографию и смежные науки. Многолетний период развития криптографии убедительно показал значительную роль различных разделов математики в создании теоретической криптографии, как науки, и разработке эффективных методов анализа и синтеза шифрсредств.

С учетом многоплановости и широкого спектра подлежащих решению сложных научно-прикладных проблем необходимо привлечь к их решению ведущих российских ученых, институты Российской Академии наук и отраслевых академий, отраслевые научно-исследовательские институты и ведущие вузы страны.

Здесь, в стенах главного учебного заведения страны, хотел бы отметить личный вклад ректора МГУ Виктора Анатольевича Садовничего в развитие проблематики информационной безопасности. По его инициативе в университете развернуты работы по широкому кругу задач обеспечения информационной безопасности как в гуманитарных, так и технических приложениях.

В соответствии с решением Ученого Совета МГУ от 9 июня 2003 г. в Московском государственном университете создан Институт проблем информационной безопасности. Основная задача Института — проведение фундаментальных и прикладных научных исследований по проблемам информационной безопасности, поэтому в его составе образованы три научных отдела:

- отдел математических проблем информационной безопасности;
- отдел информационной безопасности компьютерных систем;
- отдел гуманитарных проблем информационной безопасности.

Институт также участвует в учебно-методическом обеспечении образовательного процесса всех уровней в области информационной безопасности, в международном научном и научно-педагогическом сотрудничестве по проблемам информационной безопасности, в разработке и реализации государственных, межведомственных и международных научных программ и проектов по этой проблематике.

Во время визита в прошлом году в МГУ канцлера Университета штата Нью-Йорк доктора Роберта Л. Кинга мы договорились о проведении совместных исследований по тематике информационной безопасности. Доктор Кинг познакомил нас с программой «Кибербезопасность и защита Отечества», которая реализуется в Университете штата Нью-Йорк. За истекший год мы неоднократно встречались с нашими американскими партнерами, обсуждали направления исследований, представляющие взаимный интерес. Один из результатов этих встреч — четыре доклада американских специалистов, которые представлены на сегодняшнюю конференцию. Во время конференции будет продолжен обмен мнениями между специалистами наших университетов по определению дальнейших форм сотрудничества. В целом расширение международных контактов и преобразование статуса нашей конференции в международную характеризует объективную тенденцию к унификации подходов специалистов разных стран в обеспечении информационной безопасности в условиях новых вызовов XXI века.

Задачи, стоящие перед российским научным сообществом, по разработке систем гарантированного обеспечения информационной безопасности являются сложными, многогранными и наукоемкими. Очевидно, что никогда еще столь сильно не ощущалась необходимость совместных усилий государственных, промышленных и академических кругов по своевременному предупреждению и нейтрализации угроз информационной безопасности России и стратегических вызовов в области потенциальной информационной войны XXI века, требующих полномасштабной мобилизации всех материальных и интеллектуальных ресурсов страны.

Разрешите выразить уверенность в том, что существующие обширные научные связи между МГУ им. М. В. Ломоносова, Российской Академией наук, госучреждениями, промышленными кругами и IT-бизнес-структурами будут и в дальнейшем развиваться на благо России.

Хотелось бы пожелать всем участникам и гостям конференции эффективной и плодотворной работы, успехов в профессиональной деятельности, новых идей и практических решений проблем обеспечения информационной безопасности.

Пленарные доклады

О некоторых применениях групп в криптографии

М. М. Глухов, Б. А. Погорелов

Данный доклад не является обзором работ по применениям групп в криптографии. Его цель — обратить внимание специалистов теории групп на некоторые задачи, возникающие в приложениях групп к построению и анализу криптографических систем.

Так как шифрование информации, осуществляется, как правило, с помощью системы подстановок некоторого (шифруемого) алфавита, то естественно, что в криптографии наиболее широкое применение находят группы подстановок. Имеются шифрсистемы, в которых множество используемых для зашифрования подстановок является группой. Примером может служить система табличного гаммирования по модулю m (см. [1, стр. 126]). В ней указанные подстановки образуют регулярную группу, являющуюся группой сдвигов (трансляций) аддитивной группы кольца вычетов $\mathbb{Z}/m\mathbb{Z}$. Известны шифрсистемы, в которых используется вся симметрическая группа подстановок шифруемого алфавита или сравнительно мощное ее подмножество. К последним можно отнести шифрсистемы, реализуемые некоторыми дисковыми шифраторами (см. [1, стр. 125]). В исследуемых шифрсистемах с открытым ключом (например, типа RSA или Эль-Гамаля, см. [1, стр. 310]) используются мультипликативные группы конечных полей, группы точек эллиптических кривых, группы классов идеалов конечных алгебраических расширений поля рациональных чисел и др.

Используемые при шифровании последовательности подстановок зависят, как правило, от определенной ключевой информации.

К множеству используемых при шифровании подстановок обычно предъявляется ряд требований. В частности, желательно, чтобы подстановок было много, чтобы они обеспечивали хорошее перемешивание информации, легко реализовались и трудно восстанавливались. Перемешивающие свойства систем подстановок формализуются на теоретико-вероятностном языке, а обеспечиваются их комбинаторно-алгебраическими свойствами. При этом особую роль играют такие их свойства

Работа выполнена при поддержке гранта Президента РФ № НШ-2358.2003.9.

как транзитивность, примитивность, кратная транзитивность, мощность порождаемой ими группы и т. д.

Группы преобразований конечных множеств используются в криптографии не только для зашифрования информации, но и в различных вспомогательных целях, например, для образования ключей, для выработки управляющих последовательностей, для классификации функций и т. д.

Здесь мы укажем лишь на некоторые направления исследований, связанные с применением групп в криптографии, приведем по ним отдельные результаты и отдельные нерешенные проблемы.

Сразу отметим, что обширный материал по группам подстановок содержится в книгах [43], [27] и в обзоре [28].

Ниже симметрическая и знакопеременная группы подстановок на множестве Ω будут обозначаться соответственно через $S(\Omega)$, $A(\Omega)$, или S_n, A_n при $\Omega = \{1, \ldots, n\}$. В приложениях особую роль играет случай, когда $\Omega = \{0, \ldots, 2^m - 1\}$. В этом случае для реализации подстановок можно использовать операции $+$, $\cdot$ сложения и умножения в кольце вычетов по модулю 2^m, а также и операции $\oplus$, $\otimes$ сложения и умножения в поле $\mathrm{GF}(2^m)$ при отождествлении числа $a_{n-1}2^{n-1} + \ldots + a_1 2 + a_0$, $a_i \in \{0, 1\}$, с элементом $(a_{n-1}, \ldots, a_1, a_0)$ из $\mathrm{GF}(2^m)$.

1. Построение систем образующих симметрических и знакопеременных групп подстановок

Подстановки используются не только в криптографии. Поэтому методами их получения интересовались многие специалисты. В криптографии при получении больших массивов подстановок зачастую применяется метод образующих элементов. Например, этот метод используется в дисковых шифраторах, в стандартах США DES и AES. В связи с этим естественно возникает задача нахождения различных систем образующих симметрических и знакопеременных групп. Много конкретных результатов о системах образующих этих групп можно найти в монографии [40] и в обзорной работе [12]. В частности, в [40] доказано, что для любой неединичной подстановки g из группы S_n (за исключением случая, когда $n = 4$ и g принадлежит группе Клейна) в группе S_n найдется подстановка h, которая вместе с g порождает S_n. Об изобилии систем элементов, порождающих группу S_n, свидетельствует и теорема Дж. Диксона [39] о том, что при случайном выборе двух подстановок из группы S_n вероятность получить базис группы S_n или A_n стремится к 1 при $n \to \infty$. Аналогичный факт с оценкой скорости стремления указанной вероятности к 1 при выборе m случайных подстановок установлен в [33]. И вместе с тем, построить конкретную систему образующих с нужными свойствами для группы S_n не всегда просто. Поэтому задача построения различных легко реализуемых компьютером систем образующих групп A_n и S_n остается актуальной. Особый интерес для криптографии представляет случай $n = 2^m$. Приведем отдельные результаты в этом направлении.

1. В 1975–76 гг. П. Роулинсон и П. Вильямсон классифицировали примитивные группы подстановок степени n, содержащие 2^m-цикл при $2^m < n - 1$. В [30] рассмотрены оставшиеся (наиболее трудные) случаи, когда $2^m = n - 1, n$. В частности, доказано, что при $2^m = n$ такая группа либо содержит группу A_n, либо подстановочно изоморфна проективной линейной группе $\mathrm{PGL}(2, p)$ при простом $p = 2m - 1 > 3$.

2. Любая примитивная группа подстановок степени $n = 2^m$, являющаяся произведение двух независимых 2^{m-2}-циклов, содержит группу A_n. (А. А. Нечаев, не опубликовано).

3. С использованием результатов п.п. 1–2 в [7] доказано, что любая примитивная группа подстановок поля $P = \mathrm{GF}(2^m)$, содержащая подстановку $g = (0, 1, \ldots, 2^m - 1)$ и какое-либо аффинное преобразование h пространства $(P, \oplus)$ над полем $\mathrm{GF}(2)$, совпадает с группой $S(P)$. При этом условие примитивности легко проверяется по матрице преобразования h в стандартном базисе: в ее правом верхнем углу не должно быть нулевой подматрицы размеров $k \times (n - k)$ при $k = 1, \ldots, n$.

4. Группы, порожденные подстановками, реализуемыми за один раунд в криптосистемах DES и AES, совпадают со знакопеременной группой подстановок шифруемого (блочного) алфавита, см. [41], [42]. Вместе с тем, как показано в [38], множество всех подстановок, реализуемых в системе DES при всевозможных ключах, не является группой.

При построении конкретных систем образующих групп S_n, A_n часто используется условие примитивности группы $\langle M \rangle$, порожденной множеством M. В связи с этим для криптографии представляет интерес задача описания примитивных групп подстановок.

Примитивные группы всех степеней $n \leqslant 64$ описаны (без использования классификации конечных простых групп) в [31]. Таблица этих групп приведена также в [27]. Позднее, в работах В. И. Ильина и А. С. Токмакова, а также Дж. Диксона и Б. Мортимера с использованием классификации простых групп были описаны все примитивные группы степеней $n < 1000$ с неабелевым цоколем (библ. об этих и других результатах по указанной проблеме см. в [28]).

Вместе с тем, представляющая для криптографии особый интерес *проблема* описания примитивных групп ограниченных, но достаточно больших степеней вида 2^m, остается открытой.

2. О параметрах, связанных с заданием конечных групп системами образующих элементов

Для оценки качества систем образующих групп используются различные числовые параметры, в частности: $L(G, M)$ — длина группы G относительно системы M, т. е. минимальное натуральное число t, при котором $G = M^1 \cup \ldots \cup M^t$; $d(G, M)$ — ширина группы G относительно системы M, т. е. минимальное натуральное число d, при котором группа G представляется в виде объединения d множеств типа M^s. Для k-транзитивных групп подстановок, действующих на множестве Ω, интересным является параметр $d_k(M)$ — показатель k-транзитивности множества подстановок M, т. е. минимальное натуральное число r, при котором множество M^r k-транзитивно (если такое r существует). Грубо говоря, величины $L(G, M)$ и $d_k(M)$ характеризуют множество M с точки зрения скорости порождения группы G и скорости перемешивания элементов подстановками из G. Особенно большое число работ имеется по оценкам длин групп S_n и A_n. Обзоры этих результатов см. в [12], [17].

Здесь отметим в качестве примеров лишь следующие факты.

1. Если $G = \langle M \rangle$ — конечная группа, то последовательность ее подмножеств

$$M,\ M^2,\ M^3,\ \ldots$$

является периодической и подмножества ее минимального периода исчерпываются всеми смежными классами группы G по H, где H — минимальный нормальный делитель группы G, по которому M содержится в одном смежном классе; поэтому $d(G, M) = [G : H]$ [6].

2. Ширина конечной группы G относительно системы образующих M равна НОД длин определяющих слов в алфавите M любого задания этой группы приведенной системой определяющих соотношений [6].

3. Наилучшая из известных авторам верхних оценок длины группы S_n или A_n получена в работе [37], где доказано: если $A_n < G < S_n$, $G = \langle M \rangle$ и $M^{-1} = M$, то

$$\max L(G, M) \leqslant \exp\bigl((n \log n)^{1/2}(1 + O(1))\bigr).$$

4. Имеется ряд работ о длинах группы S_n в системе образующих $g = (1, 2, \ldots, n)$, $h = (1, 2)$. Одной из первых здесь была работа В. М. Глушкова [15] с верхней оценкой $2n^2$ длины группы. Позднее эта оценка улучшалась разными авторами. В [29] было доказано, что

$$\frac{n^2}{3} - \frac{n}{3} - 1 \leqslant L(S_n, \{g, h\}) \leqslant \frac{4n^2}{3} - \frac{n}{2} - \frac{5}{6}.$$

Совпадающие по порядку верхняя и нижняя оценки параметра $L(S_n, \{g, h\})$, давшие его асимптотику $3n^2/4$, найдены в [21].

Гипотеза: $L(S_n, M) \leqslant n^2$, если $|M| = 2$ (верна при $n < 8$, см. [32]).

5. Результаты работ [21], [29] интересны тем, что полученная в них нижняя оценка по порядку превосходит нижнюю оценку $n \log_2 n$, получаемую из мощностных соображений для всех систем из двух образующих. Систему образующих M группы G естественно назвать оптимальной по порядку, если верхняя оценка величины $L(G, M)$ совпадает по порядку с нижней мощностной оценкой для всех систем образующих из $|M|$ элементов. Первые (достаточно искусственные) примеры оптимальных по порядку k-элементных систем образующих группы S_n (при любом $k > 1$), были указаны Ю. В. Голунковым в [16]. В работах [6], [7], [35] приводятся оптимальные по порядку системы образующих группы S_n, с простой программной реализацией.

Имеющиеся примеры указывают на некоторую зависимость длины группы S_n относительно системы образующих M от показателя 2-транзитивности множества M.

Гипотеза: $L(S_n, M) \leqslant d_2(M) \cdot n$.

3. Использование регулярных групп для построения множеств подстановок с заданными свойствами

В некоторых шифрсистемах множества используемых подстановок представляются в виде композиций регулярных множеств (в частности, регулярных групп) подстановок Множество подстановок $H \subset S(\Omega)$ мы называем регулярным, если H транзитивно и $|H| = |\Omega|$. Нижние строчки табличных записей подстановок регулярного множества при одинаковых верхних строчках образуют латинский квадрат. Следовательно, указанные шифрсистемы являются композициями шифров гаммирования. К ним относятся, в частности, многие дисковые шифрсистемы, а также модификации шифрсистем DES и AES, в которых ключи каждого раунда выбираются произвольно и независимо. В каждой из этих систем составляющими регулярными множествами являются регулярные подгруппы или смежные классы по регулярным подгруппам. В связи с этим естественно возникает задача исследования композиций регулярных групп подстановок. Например, представляет

определенный интерес задача построения 2-транзитивных множеств подстановок в виде композиции возможно меньшего числа регулярных множеств и, в частности, в виде минимальной степени одного и того же регулярного множества H. Заметим, что в последнем случае соответствующий показатель степени совпадает с показателем 2-транзитивности множества H.

В общем случае известно, что вероятность 2-транзитивности произведения трех случайно выбранных регулярных множеств из S_n, стремится к единице при $n \to \infty$ (Амбросимов А. С., не опубликовано). Однако, задача построения 2-транзитивного произведения трех конкретных регулярных множеств является не простой. В [8] доказано, что 2-транзитивной является третья степень регулярного множества подстановок $\Sigma_m f h$, где Σ_m — группа сдвигов группы $(P, +)$, f — преобразование поля $P = \mathrm{GF}(2^m)$, определенное условиями $f(x) = x - 1$, если $x \neq 0$, и $f(0) = 0$, $h \in \mathrm{Aut}(P, +)$. В частности, именно такого типа множество использовано в криптосистеме AES.

В работах [6], [7] выделены классы подстановок D поля $\mathrm{GF}(2^m)$, $m > 1$, при которых 2-транзитивна 4-я степень или 5-я степень регулярного множества $\Gamma_m D$, где Γ_m — группа, порожденная подстановкой $g = (0, 1, \ldots, 2^m - 1)$.

В [9] найден показатель 2-транзитивности для множества однораундовых функций в некоторых модификациях шифрсистемы AES. Для алфавитов порядков 8^{4n}, $n = 4, 6, 8$, он оказался равным соответственно 5, 6, 7.

Ввиду сказанного представляет интерес следующий *вопрос*: существуют ли два регулярных множества, в частности, две регулярные группы G_1, G_2, произведение которых 2-транзитивно. В общем случае вопрос остается открытым Для групп имеет место следующий факт (см. [7]).

Для того, чтобы произведение двух регулярных групп подстановок было 2-транзитивным, необходимы следующие условия:

- n кратно числу вида $2^3 pq$, где p, q — различные нечетные простые числа;
- каждая из групп совпадает со своим коммутантом;
- силовская 2-подгруппа каждой из групп является обобщенной группой кватернионов.

Не известно, являются ли эти условия достаточными. Представляется вполне правдоподобной *гипотеза*: произведение GH не 2-транзитивно при любых регулярных подгруппах (и даже подмножествах) G, $H \subset S_n$.

4. О распределении вероятностей на конечных группах

Это большое самостоятельное направление. Типичными для этого направления являются задачи исследования законов распределения элементов группы на словах фиксированных длин, а также наборов элементов множества, на котором действуют подстановки группы, при заданном распределении на образующих элементах. Исследуется, в частности, предельное поведение соответствующих матриц переходных вероятностей, условия их сходимости к равномерным матрицам и сходимости к ним на конечном шаге, скорость сходимости и т. д. Заметим, что при решении указанных задач широко используется теория представлений и характеры конечных групп.

Из последних общих работ в этой области укажем статьи [18–20].

Для иллюстрации приведем один результат из работы [18]. Пусть $\xi_1, \xi_2, \ldots$ — последовательность независимых в совокупности и одинаково распределенных случайных величин на конечной группе G, $\eta(k) = \xi_1 \xi_2 \ldots \xi_k$, $M = \{g \in G : \xi_1(g) > 0\}$, $G = \langle M \rangle$, H — минимальный нормальный делитель группы G, по которому M содержится в одном смежном классе, $d = [G : H]$. Тогда последовательность $\eta(k_s)$, $k_s \in \mathbb{N}$, имеет предельное распределение в том и только том случае, когда, начиная с некоторого s, выполняется сравнение $k_s \equiv j \pmod d$ при фиксированном j. При этом предельное распределение является равномерным на подходящем смежном классе группы G по подгруппе H.

В работах [19], [20] исследуются также последовательности матриц переходных вероятностей элементов из области действия группы, соответствующих указанным выше случайным величинам $\eta(k)$, находятся оценки среднеквадратичного уклонения их элементов от элементов равномерной матрицы и условия достижимости предельных распределений на конечном шаге.

5. Группы и полугруппы изометрий

Пусть Ω — конечный алфавит, Ω^n — множество слов длины n, Ω^* — множество всех слов в алфавите Ω, $f(P, Q)$ — минимальное число искажений типа f при переходе от P к Q. Типы искажений: замена, вставка, выпадение буквы и их комбинации — всего 7 типов. Обозначим через $I(\Omega^*, f)$ полугруппу всех инъективных отображений $\varphi : \Omega^* \to \Omega^*$, не размножающих ошибок типа f, т. е. удовлетворяющих условию:

$$\forall\, P, Q \in \Omega^* \quad f(\varphi(P), \varphi(Q)) \leqslant f(P, Q), \tag{1}$$

а через $S(\Omega^*, f)$ — группу всех биекций из $I(\Omega^*, f)$, которую естественно назвать группой изометрий пространства (Ω^*, f) (хотя не каждая из функций f является метрикой).

В 1956 г. А. А. Марков полностью описал все биективные преобразования, не размножающие искажений типа замены букв, [24]. В этом случае f есть метрика Хэмминга ρ, и результат имеет вид:

$$\varphi \in S(\Omega^*, \rho) \iff \varphi(a_1 \dots a_n) = g_1(a_{h(1)}) \dots g_n(a_{h(n)}),$$

где $g_i \in S(\Omega)$, $h \in S_n$. Таким образом, в этом случае $S(\Omega^n, f) \cong S(\Omega)^n S_n$.

Заметим, что при $|\Omega| > 4$ группа $S(\Omega^n, \rho)$ является максимальной унипримитивной подгруппой симметрической группы $S(\Omega^n)$.

В последние годы в работах [4, 10, 11] были описаны полугруппы $I(\Omega^*, f)$, группы $S(\Omega^*, f)$, а также множества инъективных отображений $\varphi\colon \Omega^* \to \Omega_1^*$ с условием (1) для каждой из 7 функций f и для любых конечных Ω, Ω_1.

Проблема: описать биективные и инъективные преобразования множества Ω^*, размножающие искажения типа f не более, чем в k раз, в частности, при $k = 2$. Для искажений типа замены, т. е. для $f = \rho$, эта проблема ставилась А. А. Марковым.

В настоящее время готовится к публикации работа Б. А. Погорелова, в которой предлагается теоретико-групповой подход к построению метрик на Ω^n, «огрубляющих» метрику ρ. На этом пути для малых значениях n получены примеры групп преобразований, размножающих искажения не более, чем в 2 раза.

6. Групповая классификация функций

Пусть $F_k(n)$ — множество всех функций k-значной логики от n переменных, т. е. отображений $f\colon \Omega^n \to \Omega^n$, где $|\Omega| = k$, и G — подгруппа группы $S(\Omega^n)$. Определим действие группы G на $F_k(n)$:

$$\forall\ f \in F_k(n),\ g \in S(\Omega^n)\quad f^g(x_1, \dots, x_n) = f(g(x_1, \dots, x_n));$$

отношение эквивалентности на $F_k(n)$:

$$f_1 \sim_G f_2 \iff \exists\ g \in G\quad f_2 = f_1^g$$

и группу инерции функции f в группе G:

$$I_G(f) = \{g \in G\colon f^g = f\}.$$

В итоге множество функций разобьется на непересекающиеся классы эквивалентных функций. Порядок класса $[f]_G$, содержащего функцию f, определяется порядками групп G и $I_G(f)$:

$$|[f]_G| = |G| \cdot |I_G(f)|.$$

Проблема групповой классификации функций естественно возникает в различных приложениях функций. В криптографии в качестве Ω обычно используется кольцо или поле, а в качестве G — группы Σ_n, S_n, $S(\Omega)n$, $\mathrm{GL}(n, \Omega)$, $\mathrm{AGL}(n, \Omega)$ и их произведения, где Σ_n — группа сдвигов группы $(\Omega^n, +)$; S_n — группа перестановок переменных. При этом представляют интерес задачи нахождения представителей классов $[f]_G$, групп инерции заданной функции f и построение функций с заданной группой $I_G(f)$. Обзор ранних результатов по этой тематике (до 1988 г.) можно найти в [13]. Известно [14], что для любой конечной группы H и любого $k \in \mathbb{N}$ существуют такие n и $f \in F_k(n)$, что $I_G(f) \cong H$ при $G = S_n$. Хорошо известна теорема Шеннона о тривиальности группы инерции для почти всех функций $f \in F_k(n)$ в группе $\Sigma_n \cdot S_n$ при $n \to \infty$. Имеются ее обобщения на группу $\mathrm{AGL}(n, \Omega)$, когда Ω — конечное поле (см. [22]) и когда $\Omega = \mathbb{Z}/m$ (см. [14]). В работе [2] она обобщена на группу $\mathrm{AGL}(n, R) \cdot H$, где R — любое конечное коммутативное кольцо, H — любая полурегулярная группа. При этом найдены асимптотические разложения числа функций с нетривиальной группой инерции и для числа классов эквивалентности. Несмотря на указанный эффект Шеннона, задача классификации функций при фиксированных n, k и G является сложной, особенно для линейной и аффинной групп. Из последних достижений в этом направлении укажем на результаты А. В. Черемушкина, полученные в 1997 г. и опубликованные в [34]. Им для булевых функций проведена классификация кубических форм от 6, 7, 8 переменных с точностью до квадратичной части, всех функций от 6 переменных с точностью до кубической части, функций 4-й степени с точностью до квадратичной части. При этом автором найдены представители классов и вычислены группы инерции функций. Заметим, что независимо классификация булевых кубических форм от 7 и 8 переменных с точностью до квадратичной части проведена в [44].

В приложениях зачастую сложную функцию от большого числа переменных получают в виде суперпозиции или итерации функций от меньшего числа переменных. В связи с этим представляется интересной *проблема* о связях группы инерции полученной функции с группами инерции исходных функций.

7. Решение уравнений в группах

В некоторых шифрсистемах для зашифрования информации используются подстановки, представимые в виде произведения известных и неизвестных (ключевых) подстановок. В связи с этим для криптографии представляет интерес задача о решении в конечных группах

уравнений вида

$$A_1X_1\ldots A_nX_nA_{n+1}=B, \tag{2}$$

где $A_1, \ldots, A_{n+1}$ — известные, а $X_1, \ldots, X_n$ — неизвестные элементы некоторой группы G.

Задача решения уравнений в группах возникает не только в криптографии, но и в других приложениях теории групп, да и при изучении самих групп. Поэтому уравнениями в группах занимались многие известные специалисты теории групп — Ф. Г. Фробениус, М. Холл, В. Магнус и др. В принципе, задача нахождения всех решений уравнения (2) в конечной группе G алгоритмически разрешима, например, полным перебором всех систем $(X_1, \ldots, X_n)$ элементов из G. Однако при постановке указанной задачи в криптографии речь идет о поиске практически приемлемых алгоритмов.

В общем случае задача нахождения простых алгоритмов решения уравнений вида (2) является сложной, и потому решалась она лишь в отдельных частных случаях. Так, в [36] она решена для нильпотентных групп, в [3] — для разрешимых групп (с использованием модифицированного варианта дифференциального исчисления Фокса). Для симметрической группы известны простые методы решения уравнений вида $X^d=1$, $X^{-1}gX=h$ и некоторых других частных типов.

В ряде криптосистем с открытым ключом задача нахождения секретного ключа сводится к решению уравнений вида $a^x=b$ в конечных абелевых группах. Последняя задача называется задачей дискретного логарифмирования в соответствующей группе. В последние годы различными авторами было предложено большое число алгоритмов решения этой задачи. Обзор и библиографию по этим методам можно найти в [5]. Отметим, что в настоящее время асимптотически наилучшим по трудоемкости является метод решета числового поля. Он имеет субэкспоненциальную сложность. Для мультипликативной группы простого поля $\mathrm{GF}(p)$ асимптотическая оценка сложности этого метода имеет вид $\exp\bigl((c+o(1))(\ln\,p)^{1/3}(\ln\ln\,p)^{2/3}\bigr)$. В последние годы борьба идет за уменьшение константы c. Укажем в связи с этим на работы [25], [26], в которых предложена и обоснована новая модификация метода решета числового поля в применении к задаче дискретного логарифмирования в поле $\mathrm{GF}(p)$. При этом константа c оказалась равной 1,902 и наилучшей из известных (на время публикации работы [26]).

Заметим, что в методе решета числового поля для решения уравнения в группе используется опять-таки группа, а именно — группа классов дробных идеалов конечного алгебраического расширения поля рациональных чисел.

Список литературы

[1] *Алферов А. П., Зубов А. Ю., Кузьмин А. С., Черемушкин А. В.* Основы криптографии. М., Гелиос АРВ, 2001.

[2] *Амбросимов А. С., Шаров Н. Н.* О некоторых асимптотических разложениях для числа функций с нетривиальной группой инерции. Проблемы кибернетики, вып. 36, 1979.

[3] *Анашин В. С.* Разрешимые группы с операторами и коммутативные кольца, обладающие транзитивными полиномами. Алгебра и логика, т. 21, № 6, 1982, 627–646.

[4] *Бабаш А. В., Глухов М. М., Шанкин Г. П.* О преобразованиях множества слов в конечном алфавите, не размножающих искажений. Дискр. м., т. 9, вып. 3, 1997, 3–19.

[5] *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. М., МЦНМО, 2003.

[6] *Глухов М. М.* О числовых параметрах, связанных с заданием конечных групп системами образующих элементов. Труды по д. м., т. 1, 1997, 43–66.

[7] *Глухов М. М.* О 2-транзитивных произведениях регулярных групп подстановок. Труды по д. м., т. 3, 2000, 37–52.

[8] *Глухов М. М.* О матрицах переходов разностей для некоторых классов преобразований. Обозрение ППМ, т. 10, вып. 3, 2003, 634.

[9] *Глухов М. М.* О матрице частот переходов пар блоков в одной модификации криптосхемы «Rijndael». Обозрение ППМ, т. 11, вып. 2, 2003, 634.

[10] *Глухов М. М.* Инъективные отображения слов, не размножающие искажений типа пропуска букв. Дискр. м., т. 11, вып. 2, 1999, 20–39.

[11] *Глухов М. М.* Инъективные отображения слов, не размножающие искажений. Труды по д.м., т. 4, 2001, 17–32.

[12] *Глухов М. М., Зубов А. Ю.* О длинах симметрических и знакопеременных групп подстановок в различных системах образующих (обзор). Матем. вопр. кибернетики, вып. 8, 1999, 5–32.

[13] *Глухов М. М., Ремизов А. Б., Шапошников В. А.* Обзор по теории k-значных функций. Часть I. М., 1988.

[14] *Глухов М. М.* Математическая логика (уч. пособие), М, 1981.

[15] *Глушков В. М.* О полноте систем операций в электронных вычислительных машинах. Кибернетика, № 2, 1968, 1–5.

[16] *Голунков Ю. В.* О сложности представления подстановок симметрической полугруппы через элементы систем образующих. Кибернетика, № 1, 1971, 43–44.

[17] *Голунков Ю. В.* Алгоритмическая полнота и сложность микропрограмм. Кибернетика, № 3, 1977, 1–15.

[18] *Горчинский Ю. Н., Круглов И. А., Капитонов В. М.* Вопросы теории распределений на конечных группах. Труды по д. м., т. 1, 1997, 85–112.

[19] *Горчинский Ю. Н.* О средних квадратических уклонениях матриц перехода на конечных группах подстановок четного порядка. Труды по д. м.,

т. 3, 2000, 73–94.
[20] *Горчинский Ю. Н.* Об улучшении оценок средних квадратических уклонений матриц перехода произведения независимых случайных величин на конечных группах подстановок. Труды по д.м., т. 3, 2000, 53–72.
[21] *Зубов А. Ю.* О диаметре группы S_n относительно системы образующих, состоящей из полного цикла и транспозиции. Труды по д. м., т. 2, 1998, 112–150.
[22] *Клосс Б. М.*, *Нечипорук Э. И.* О классификации функций многозначной логики. Проблемы кибернетики, вып. 9, 1963, 63–98.
[23] *Логачев О. А.*, *Сальников А. А.*, *Ященко В. В.* Булевы функции в теории кодирования и криптологии. М., МЦНМО, 2004.
[24] *Марков А. А.* О преобразованиях, не распространяющих искажений. Избранные труды, т. 2, 70–93.
[25] *Матюхин Д. В.*, *Мурашов Н. Н.* Модификация метода решета числового поля для дискретного логарифмирования в поле $\mathrm{GF}(p)$. Обозрение ППМ, т. 7, вып. 2, 2000, 387–389.
[26] *Матюхин Д. В.* Об асимптотической сложности дискретного логарифмирования в поле $\mathrm{GF}(p)$. Дискр. м., т. 15, вып. 1, 2003, 28–49.
[27] *Погорелов Б. А.* Основы теории групп подстановок. Часть 1. Общие вопросы. М., 1986.
[28] *Погорелов Б. А.* Группы подстановок. Часть 1. (Обзор за 1981–95 гг.). Труды по д. м., т. 2, 1998.
[29] *Погорелов Б. А.* Об одной задаче В. М. Глушкова. Кибернетика, № 5, 1973, 141–144.
[30] *Погорелов Б. А.* Примитивные группы подстановок, содержащие 2^m-цикл. Алгебра и логика, т. 19, № 2, 1973, 236–247.
[31] *Погорелов Б. А.* Примитивные группы подстановок малых степеней. Алгебра и логика, т. 19, № 3, 1973, 423–457.
[32] *Рубцов В. А.* О некоторых оценках меры информационной избыточности систем образующих, порождающих симметрическую группу подстановок. Кибернетика, № 5, 1975, 51–55.
[33] *Сачков В. Н.* Группы подстановок и полугруппы преобразований с заданным числом образующих. Труды по д. м., т. 3, 2000, 215–234.
[34] *Черемушкин А. В.* Методы линейной и аффинной классификации функций. Труды по д. м., т. 4, 2001, 273–314.
[35] *Шапошников И. Г.* О некоторых системах образующих симметрической и знакопеременной групп, допускающих простую программную реализацию. Дискр. м., т. 16, вып. 1, 114–120.
[36] *Шмелькин А. Л.* О полных нильпотентных группах. Алгебра и логика, т. 6, № 2, 1967, 111–113.
[37] *Babai L.*, *Seress A.* On the diameter of Cayley graphs of the symmetric group. J. comb. Theory, v. A49, 1988, 175–179.
[38] *Cambell K.*, *Wiener M.* DES is not a group. Crypto'92, 512–520.
[39] *Dixon J. D.* The probability of generating the symmetric groups. Mat. Z., v. 110, 1969, 199–205.
[40] *Piccard S.* Sur les bases des groupes d'ordre fini. Neuchatel, 1957.
[41] *Wernsdorf R.* The one-round function of the DES generate alternating group. Eurocrypt'92, 99–112.
[42] *Wernsdorf R.* The round function of RIJNDAEL generate the alternating groups. Fast software 2002, Leuven, Berlin, Preproceed., 272.
[43] *Wielandt H.* Finite permutation groups. N.Y., 1964.
[44] *Xiang-Dong Hou.* $\mathrm{GL}(m, 2)$ acting on $R_m/R(m)$. J. Discr. Math., v. 149, 1996, 99–122.

Современные направления стеганографии

Н. П. Варновский, Е. А. Голубев, О. А. Логачёв

1. Введение

Стеганография представляет собой специфическую область человеческой деятельности, связанную с разработкой и анализом методов сокрытия факта передачи информации. Подобно криптографии, стеганография известна со времен античности. Но на этом аналогии, по крайней мере в контексте теоретических исследований, заканчиваются. За последние четверть века возникла и успешно развивается новая математическая дисциплина — криптология, или, что то же самое, математическая криптография, изучающая математические модели криптографических схем. Попытки создания математической стеганографии (которую, быть может, следует именовать стеганологией) предпринимаются, но исследования здесь находятся лишь в зачаточном состоянии.

Такое положение дел обусловлено прежде всего сложностью возникающих в стеганографии задач. Всякая попытка построения математических моделей стеганографических систем сопряжена с необходимостью рассмотрения большого количества случаев и подслучаев, не допускающих простой и единообразной трактовки. Другими словами, внешняя среда, в которой должны функционировать стеганографические системы, имеет гораздо большее, по сравнению с внешней средой криптографических схем, количество степеней свободы.

Но, тем не менее, теоретические исследования в области стеганографии ведутся и основная цель настоящей работы состоит в изложении некоторых результатов этих исследований.

Заголовок статьи следует понимать как сокращение. Полное название должно было бы звучать примерно так: «Современное состояние математических исследований в стеганографии. Основные направления, модели и понятия».

2. Модели стеганографических каналов

Впервые в открытой литературе модель стеганографического канала была описана Симмонсом в работе, представленной на конференцию Crypto'83 [14]. Из-за существовавших в то время запретов на публикации по стеганографии Симмонс сформулировал эту модель на языке задачи о двух заключенных.

Двое заключенных, Алиса и Боб, находящиеся в различных тюремных камерах, могут обмениваться посланиями. Но вся их переписка проходит через руки тюремного надзирателя Уэнди. Алиса и Боб должны выработать план побега, обмениваясь внешне безобидными текстами или картинками, не вызывающими подозрения у противника (Уэнди).

В соответствии с терминологией, согласованной на конференции Information Hiding: First International Workshop [13], те сообщения, которые Алиса и Боб пытаются передать друг другу втайне от противника, называются *встроенными* (*embedded*) *сообщениями*. Предлагается также обобщенный термин *встроенное сообщение* ⟨*тип данных*⟩, где ⟨*тип данных*⟩ заменяется, как и всюду ниже, соответствующим типом данных: текст, изображение (image) и т. п. Однако, по нашим наблюдениям, этот термин не прижился и вместо него обычно используется термин *скрытое* (*hidden*) *сообщение*. Тип данных обычно ясен из контекста и в большинстве случаев может быть опущен.

В стеганографической литературе зачастую предполагается, что сообщение, которое Алиса пытается передать Бобу, предварительно шифруется. В этом случае под скрытым сообщением понимается криптограмма.

Та информация, в которую встраивается скрытое сообщение, называется *контейнером* (*cover*, *carriage*). Обобщенный термин — *контейнер* ⟨*тип данных*⟩. Соответственно, различают текстовые контейнеры, аудиоконтейнеры, видеоконтейнеры и т. п.

Если у Алисы и Боба имеется некоторая общая секретная информация, то последняя называется *секретным стегоключом* или просто *секретным ключом*. Заметим, что наличие у Алисы и Боба общего секретного ключа не является необходимым требованием для создания стеганографического канала. Существует стеганография с открытым ключом [2], [5] и даже так называемая чистая (безключевая) стеганография [5]. В настоящей статье эти два варианта не рассматриваются.

Для контейнера, содержащего встроенное в него скрытое сообщение, используется термин *стего* ⟨*тип данных*⟩ или просто *стего*. Контейнер, не содержащий скрытого сообщения, будем называть пустым контейнером. Отметим, что многие авторы используют термин стего для того контейнера, который Алиса передает Бобу, вне зависимости от того, является ли он и в самом деле стего или пустым контейнером. Чтобы избежать противоречия, можно принять соглашение, в соответствии с которым пустой контейнер — это стего с пустым скрытым сообщением.

Совокупность тех средств, которые Алиса и Боб используют для создания стеганографического канала, называется *стеганографической системой* или просто *стеганосистемой*. Здесь сказывается влияние криптографии и принятой в ней традиции выделять криптосистемы в отдельный тип. Правильнее было бы говорить о стеганографических протоколах.

Стеганосистема определяется прежде всего теми преобразованиями, Emb и Ext, которые Алиса и Боб используют соответственно для встраивания скрытого сообщения в контейнер и для извлечения этого сообщения из стего. Более формально, Алиса вычисляет

$$\text{стего} = \text{Emb}(\text{контейнер, скрытое сообщение, ключ}),$$

а Боб —

$$\text{скрытое сообщение} = \text{Ext}(\text{стего, ключ}).$$

При этом требуется, чтобы при любом данном ключе Боб с достаточно большой вероятностью извлекал из стего в точности то скрытое сообщение, которое было встроено в него Алисой.

Преобразование Emb является, вообще говоря, не всюду определенным. Для простоты изложения доопределим его, полагая для всякой тройки (контейнер, скрытое сообщение, ключ), не входящей в область определения, значение преобразования Emb равным пустой строке λ.

Существенным элементом описания стеганографической системы является предположение об источнике контейнеров. Возможны следующие три варианта:

- контейнер поступает извне. Это означает, что Алиса никак не влияет на его выбор. Своего рода предельный случай возникает, когда контейнер выбирается противником;
- Алиса выбирает контейнер из некоторого множества контейнеров. На практике, однако, это множество не имеет обычно сколько-нибудь точного описания, что и составляет одну из основных проблем построения математической теории стеганографических систем;
- контейнер генерируется стеганосистемой в процессе вычисления преобразования Emb. Фактически это означает, что контейнер здесь вообще ни при чем, а стеганосистема по скрытому сообщению и ключу сразу создает стего.

Легко понять, что эти предположения об источнике контейнеров являются, по существу, предположениями о противнике. Основное предположение о противнике состоит в том, что он будет пропускать по каналу связи те контейнеры, которые покажутся ему естественными.

Проблема создания математической модели естественного контейнера является важнейшей исследовательской проблемой в стеганографии. На данный момент не известно даже, возможно ли в принципе создание такой модели.

В литературе по стеганографии упоминаются следующие три типа противников:

- *пассивный*. Уэнди действует так, как это описано в классической модели Симмонса;
- *активный*. Уэнди может вносить небольшие изменения в передаваемый контейнер. Как правило, предполагается, что ее вмешательство должно оставаться незаметным для Боба;
- *злоумышленный*. На действия Уэнди не накладывается никаких ограничений. Она может как угодно изменять контейнеры или даже вообще блокировать канал связи между Алисой и Бобом. Разумеется, от злоумышленного противника невозможно защититься. Но в реальности обычно имеется множество внешних обстоятельств, препятствующих злоумышленному поведению Уэнди.

Отсутствие четких критериев, отграничивающих активных противников от злоумышленных, — еще одна серьезная проблема, возникающая при построении математической теории стеганографических систем.

Противник может считаться активным также и в том случае, когда он проводит активную атаку на стеганосистему: атаку с выбором скрытого сообщения, атаку с выбором контейнера и т. п. (см. раздел 3).

Помимо термина «стеганографический канал» в литературе встречаются еще два: *скрытый* (*covert*) и *подпороговый* (*subliminal*) каналы. Поскольку в стеганографии еще нет общепринятой терминологии, возможно, что разные авторы вкладывают различный смысл в эти термины. Ниже мы приводим трактовки, которые нам удалось синтезировать на основе анализа работ по стеганографии.

Скрытый канал — это стеганографический канал, все внешнее поведение которого полностью описывается контейнерами. Иными словами, множество всех допустимых стего совпадает со множеством всех пустых контейнеров. Формально, стеганосистема для скрытого канала должна удовлетворять следующему требованию. Пусть C — множество всех контейнеров, K — множество ключей, M — множество скрытых сообщений и $S = \{s = \text{Emb}(c, m, k),\ s \neq \lambda \mid c \in C,\ m \in M,\ k \in K\}$ — множество стего. Тогда $S \subseteq C$. Стеганографические каналы, создаваемые в операционных системах и других системах программного обеспечения, а также в криптографических протоколах, во многих слу-

чаях могут быть только скрытыми. Те из стеганографических каналов, которые не являются скрытыми, мы в дальнейшем будем называть общими.

Термин *подпороговый канал* был введен Симмонсом в связи с исследованием возможности создания скрытых каналов в криптографических протоколах (аутентификации, электронной подписи). По-видимому, он хотел подчеркнуть следующую особенность таких каналов. В обычном стеганографическом канале скрытое сообщение можно запрятать в один из «углов» контейнера в надежде, что противник не станет «шарить» по всем углам. Но если он посмотрит в нужный угол, то с большой вероятностью обнаружит скрытое сообщение. Криптографические протоколы позволяют строить скрытые каналы, существование которых, в предположении стойкости (криптографической) самого протокола, невозможно обнаружить в принципе. Для таких скрытых каналов целесообразно ввести специальный термин — подпороговый канал. В качестве примера можно рассмотреть вырожденный случай, когда Уэнди разрешает пересылку любых случайных последовательностей битов. Тогда шифр Вернама обеспечивает абсолютно необнаружимый скрытый канал. Термин «подпороговый» (subliminal) заимствован из психологии и означает, в развернутой формулировке, «находящийся ниже границы восприятия».

Возможно также, что единственной причиной появления словосочетания «подпороговый канал» послужила необходимость подыскать синоним, который мог бы заменить термин «стеганографический канал» в открытых публикациях.

При построении стеганосистем для скрытых каналов у стеганографа остается меньше свободы, поэтому данная задача в определенном смысле сложнее задачи построения стеганосистем для общего стеганографического канала. Но в то же время и возможности активного противника оказываются сильно ограниченными. Например, если речь идет о подпороговом канале в протоколе аутентификации, то активный противник может вносить в пересылаемые данные только такие изменения, которые не препятствуют выполнению протокола аутентификации в соответствии с его криптографическим определением.

Одной из важнейших характеристик стеганографического канала является его пропускная способность, под которой понимается максимальная длина скрытого сообщения, пересылаемого в одном контейнере. Существует вполне очевидное с интуитивной точки зрения, но не поддающееся формализации, разделение стеганографических каналов на *широкополосные* и *узкополосные*. Все возникающие в стеганографии научно-технические проблемы относятся к случаю широкополосных каналов. Если один или несколько битов скрытого сообщения передаются в огромном потоке, скажем аудио- или видеоинформации, то ясно, что такой стеганографический канал будет практически необнаружим. Но здесь возникает терминологическая проблема: где проходит граница между стеганосистемами для узкополосных каналов и системами передачи условных знаков. Например, Алиса должна передать Бобу всего один бит информации. Они заранее договорились, что Алиса будет пересылать сводку погоды, и если в этой сводке будет слово «облачно», то значение бита равно 0, а если слово «пасмурно», то оно равно 1. Являются ли системы передачи условных знаков предметом стеганографии? По этому поводу нет единого мнения даже у авторов настоящей статьи.

Так же как и в случае криптосистем, стойкость стеганосистемы (стеганостойкость) может быть определена только относительно конкретной пары (атака, угроза). В разделе 3. рассматриваются угрозы безопасности стеганографических систем и обсуждается классификация атак на стеганосистемы.

Здесь уместно сделать два важных замечания.

1. По аналогии с подходом, принятым в криптографии, авторы теоретических работ по стеганографии предполагают, что стеганосистема известна противнику полностью, за исключением секретного ключа. Как и в криптографии, имеются два основных довода в пользу принятия такого предположения. Во-первых, без него не ясен сам предмет исследования. Во-вторых, всегда разумно обеспечить некоторый запас прочности, сделав предположение в пользу противника. Но все же для стеганографии это предположение представляется более далеким от реальности, чем для криптографии.

2. Говоря о противнике, авторы работ по стеганографии не уточняют, о каком противнике идет речь. Возможны, по крайней мере, два принципиально различных варианта. В первом из них Уэнди осуществляет выборочный беглый просмотр большого объема информации, пересылаемой, например, в компьютерной сети большим количеством абонентов. Во втором Уэнди прослушивает один канал связи между Алисой и Бобом и может потратить значительные усилия на выявление в нем стеганографического канала. Даже весьма поверхностного анализа литературы достаточно, чтобы понять, что авторы работ, посвященных синтезу стеганографических систем, обычно подразумевают первый вариант, а авторы работ по их анализу — второй.

Специалисты, занимающиеся анализом стеганосистем с целью обнаружения слабостей в их конструкциях, называются *стеганалитиками*, а область их деятельности — *стеганализом*.

3. Атаки на стеганографические системы и угрозы их безопасности

Ниже мы перечисляем основные угрозы безопасности стеганографических систем.

Обнаружение стеганографического канала. Это — самая слабая из угроз безопасности стеганосистем. Она может быть осуществлена пассивным противником. Сама семантика слова «стеганография» требует признать стеганосистему нестойкой, если противник может обнаружить создаваемый ею стеганографический канал. Поэтому о защите от этой угрозы часто говорят как об основной задаче стеганографии. Заметим, однако, что в большинстве работ не уточняется, что понимается под обнаружением стеганографического канала. Здесь имеются две возможности:

- передается последовательность контейнеров и либо все они пустые, либо часть из них являются стего, созданными с помощью одной и той же стеганосистемы (такой стеганографический канал называется каналом с повторениями). Если не все контейнеры пустые, то противник должен рано или поздно установить этот факт;
- передается один контейнер и противник должен распознать, содержит ли этот контейнер встроенное сообщение (канал без повторений).

Возможен и такой сценарий. Уэнди получила некоторую последовательность контейнеров, которые Алиса пересылала Бобу. В результате анализа этих контейнеров Уэнди могла установить наличие стеганографического канала и даже полностью или частично прочитать скрытые сообщения. После этого Алиса передает еще один контейнер. Требуется выяснить, содержит ли он скрытое сообщение. Но подобные вариации определяются уже не столько угрозой безопасности стеганосистемы, сколько типом атаки на нее (см. ниже).

Подчеркнем, что во всех вариантах задача противника состоит в том, чтобы отличить стего от пустого контейнера.

Извлечение скрытого сообщения. Противник должен найти скрытое сообщение, содержащееся в данном стего. Существует и более слабый вариант этой угрозы: противник должен получить какую-либо частичную информацию о скрытом сообщении. Эта угроза также может быть осуществлена пассивным противником.

Разрушение скрытого сообщения. Такая угроза существует только со стороны активного противника. В этом случае Уэнди должна внести в контейнер такие допустимые изменения, чтобы в результате

Боб не смог извлечь из него скрытое сообщение (если таковое в нем было).

Подмена скрытого сообщения. Это — самая сильная из угроз; она может быть осуществлена только активным противником. Содержащееся в контейнере скрытое сообщение Уэнди должна заменить другим, выгодным ей, сообщением так, чтобы Боб не заподозрил подмену.

Существуют две основные характеристики угроз безопасности стеганографических систем. Сила угрозы определяется тем, насколько серьезными могут быть последствия ее осуществления для Алисы и Боба. Как уже отмечалось выше, подмена скрытого сообщения — самая сильная из угроз. Что же касается сложности осуществления угроз, то здесь ситуация не настолько простая, как это может показаться на первый взгляд. В литературе нередко можно встретить утверждения, что активный противник имеет значительное преимущество в том смысле, что разрушить скрытое сообщение значительно проще, чем обнаружить стеганографический канал. Например, если скрытое сообщение пересылается в младших битах пикселей изображения, то активному противнику достаточно заменять эти биты во всех контейнерах (независимо от того, являются ли они пустыми) на случайные. Однако, существуют сценарии (см. подраздел 5.3), в которых активный противник не имеет существенных преимуществ.

Известны следующие основные типы атак на стеганографические системы (см. [11]).

Атака с известным стего. Самая слабая из всех возможных атак, которую всегда может провести пассивный противник. В случае стеганографического канала без повторений предполагается, что в распоряжении Уэнди имеется только контейнер, который Алиса передавала Бобу. На основе анализа этого контейнера Уэнди должна осуществить свою угрозу безопасности стеганосистемы (обнаружить, извлечь, разрушить, подменить). Для стеганографического канала с повторениями возможны два варианта этой атаки:

1. Уэнди получает некоторую последовательность контейнеров, пересылаемых Алисой Бобу. Если среди этих контейнеров имеются стего, то предполагается, что все они созданы с помощью одной и той же стеганосистемы. Если угрозой является обнаружение стеганографического канала, то данная атака очевидным образом сводится к предыдущему случаю (одного контейнера). Для остальных типов угроз ситуация несколько сложнее, но также позволяет перейти, при подходящей переформулировке угрозы, к случаю одного контейнера.

2. Уэнди получает последовательность $C = \{c_1, \ldots, c_n\}$ контейнеров, пересылаемых Алисой Бобу. Кроме того, Уэнди известно, что все кон-

тейнеры из некоторого подмножества $C_1 \subseteq C$ пустые, а контейнеры из $C_2 \subseteq C$ являются стего. Как и прежде предполагается, что все стего из C, а также контейнер c_{n+1}, если он не пустой (см. ниже), созданы одной и той же стеганосистемой. Уэнди получает также информацию, частичную или полную, о скрытых сообщениях, которые содержатся в стего из множества $C_3 \subseteq C_2$. Далее Уэнди получает контейнер c_{n+1} и должна на основе его анализа осуществить угрозу безопасности стеганосистемы. Принципиальное отличие от случая 1 в том, что угроза безопасности, какой бы она ни была, относится только к контейнеру c_{n+1}.

Атака с известным контейнером. Уэнди получает контейнер, пересылаемый Алисой Бобу, и соответствующий ему пустой контейнер. Детерминированный случай тривиален. Более содержателен следующий сценарий. В каждый контейнер перед передачей по каналу связи между Алисой и Бобом вносятся небольшие случайные искажения. Уэнди знает исходный пустой контейнер и контейнер, передаваемый по каналу связи. Ее основная задача — понять, что содержится в последнем — случайный шум или скрытое сообщение.

Естественным усилением данной атаки является атака с выбором контейнера, когда Уэнди имеет возможность выбрать исходный пустой контейнер.

Атака с известным скрытым сообщением. Эта атака возможна только в случае стеганографического канала с повторениями и может быть осуществлена пассивным противником. Предполагается, что Уэнди знает стего и, быть может, соответствующий пустой контейнер. Кроме того, она каким-то образом узнает скрытое сообщение, встроенное в стего, и использует эту информацию для анализа стеганосистемы (например, пытается определить секретный ключ, чтобы реализовать какие-либо угрозы безопасности стеганосистемы в будущем).

Атака с выбором скрытого сообщения. Аналогична предыдущей, но противник может сам выбирать скрытое сообщение. Разумеется, такую атаку может провести только активный противник. Возможен, например, следующий сценарий. Уэнди «подбрасывает» нужное ей сообщение Алисе и, получив стего, пытается определить секретный ключ стеганосистемы.

4. Теоретико-информационный подход

В настоящем разделе рассматривается теоретико-информационный подход к определению стойкости стеганосистем для стеганографических каналов без повторений в присутствии пассивного противника, обладающего неограниченными вычислительными возможностями.

В основе всех известных определений стойкости стеганосистем лежит требование неотличимости распределения вероятностей на множестве стего от распределения вероятностей на множестве пустых контейнеров. В данном разделе рассматривается статистическая неотличимость, или, иначе говоря, неотличимость относительно произвольных алгоритмов. В разделе 5.исследуется стойкость, определяемая вычислительной неразличимостью, т. е. неотличимостью относительно алгоритмов с ограничениями на вычислительные ресурсы.

Парадигма неотличимости распределений вероятностей заимствована из математической криптографии. Заметим, однако, что ее адекватность для стеганографии не очевидна. По крайней мере, в случае стеганографического канала без повторений не ясно, насколько оправданными будут усилия отправителя по имитации распределения вероятностей на множестве пустых контейнеров. Не следует ли вместо этого стремиться передать скрытое сообщение в одном из наиболее вероятных контейнеров?

Всюду далее в данной работе используется следующая терминология. У отправителя имеются контейнер, называемый исходным, а также функция (алгоритм) G, позволяющая преобразовать исходный контейнер в онтейнер с шумом. Например, в случае контейнеров, содержащих визуальную информацию, существует некоторый объект, такой как картина, пейзаж и т. п. Исходный контейнер — это электронная фотокопия объекта. Алгоритм G позволяет вносить в эту фотокопию случайные искажения, соответствующие небольшим изменениям уровня освещенности, угла, под которым камера направлена на объект и т. п. Очевидна идея создания стеганографического канала путем маскировки скрытого сообщения под шум, вносимый алгоритмом G в исходный контейнер. Предполагается, что Уэнди с большой вероятностью пропускает по каналу связи пустые контейнеры с шумом, создаваемые алгоритмом G. Поэтому последние называются также допустимыми контейнерами.

4.1. Модель стеганосистемы

Пусть $\mathcal{C}$, $\mathcal{K}$ и $\mathcal{M}$ — некоторые конечные множества. В рассматриваемой модели контейнеры (как исходный, так и контейнер с шумом), секретный ключ и скрытое сообщение являются случайными величинами на множествах $\mathcal{C}$, $\mathcal{K}$ и $\mathcal{M}$ соответственно. Мы будем обозначать через C, K и M случайные величины, являющиеся соответственно исходным контейнером, секретным ключом и скрытым сообщением.

Обозначим через R случайную величину на некотором конечном множестве $\mathcal{R}$. Эта случайная величина используется Алисой для внесения дополнительной случайности в стего.

В рассматриваемой модели Алиса и Боб имеют секретный ключ k, являющийся значением случайной величины K. Чтобы передать Бобу сообщение m (являющееся значением случайной величины M), Алиса вычисляет стего $s = \mathrm{Emb}(c, m, k, r)$, где c и r — значения случайных величин C и R соответственно, и посылает его Бобу. Без ограничения общности можно считать, что $s \in \mathcal{C}$. Получив стего s, Боб может восстановить m путем вычисления $\mathrm{Ext}(s, k)$. Алиса также может послать Бобу значение случайной величины $G(c, D)$, где D — случайная величина на некотором конечном множестве $\mathcal{D}$, а G — некоторая функция из $\mathcal{C} \times \mathcal{D}$ в $\mathcal{C}$. Таким образом, D играет роль шума.

Противник (Уэнди) в рассматриваемой модели представляет собой семейство функций $\{A_c | c \in \mathcal{C}\}$, где $A_c \colon \mathcal{C} \to \{0, 1\}$. Цель противника состоит в том, чтобы, зная значение C, отличить стего (значение $S = \mathrm{Emb}(C, M, K, R)$) от контейнера с шумом (значения $C' = G(C, D)$). Если c — известный противнику исходный контейнер, то равенство $A_c(x) = 1$ ($A_c(x) = 0$) означает, что, по мнению противника, контейнер $x \in \mathcal{C}$ является стего (соответственно, является пустым).

Таким образом, рассматривается стойкость стеганосистемы против угрозы обнаружения стеганографического канала на основе атаки с известным контейнером.

Вышеуказанный противник может делать ошибки двух типов, называемые в математической статистике ошибками первого и второго рода. *Ошибкой первого рода* считается ситуация, когда Уэнди приняла пустой контейнер за стего. Если же Уэнди посчитала стего пустым контейнером, то имеет место *ошибка второго рода*. Обозначим через α и β средние (по C) вероятности ошибок первого и второго рода соответственно.

4.2. Определения стойкости

Из литературы известны две попытки определить теоретико-информационную стойкость стеганосистем. Определение Кашена [4] основано на следующем требовании: энтропия пустого контейнера (контейнера с шумом) относительно стего должна быть мала. Подчеркнем, что речь идет об относительной (relative) энтропии; см. определение ниже. Таким образом, Кашен рассматривает задачу противника по различению пустого контейнера и стего как задачу проверки статистических гипотез. Другой подход описан в работе Цёлльнера и др. [15]. Он основан на таком требовании: знание контейнера и соответствующего ему стего не уменьшает энтропию скрытого сообщения. Заметим, что здесь под задачей противника по существу понимается извлечение некоторой

информации о скрытом сообщении (очевидно, что обнаружение стеганографического канала является частным случаем этой задачи).

Известна также работа Андерсона и Птиколя [3] и ее ранняя версия [2], в которых формулируются некоторые математические утверждения (например, оценка емкости стеганографических каналов сверху через разность энтропий). Однако эти работы являются обзорными и не дают математически строгих определений рассматриваемых понятий.

Пусть X, Y и Z — случайные величины с конечными носителями $\mathcal{X}$, $\mathcal{Y}$ и $\mathcal{Z}$ соответственно. Через $\mathrm{Ent}(X)$ обозначается шенноновская энтропия случайной величины X. Пусть также $\mathrm{Ent}(X|Y)$ — условная энтропия X при условии Y. Хорошо известно, что $\mathrm{Ent}(X|Y) \leqslant \mathrm{Ent}(X)$, причем $\mathrm{Ent}(X|Y) = \mathrm{Ent}(X)$ тогда и только тогда, когда X и Y независимы. В качестве меры близости распределений случайных величин X и Y будет использоваться *энтропия X относительно Y*, обозначаемая через $\mathrm{REnt}(X\|Y)$ и определяемая формулой

$$\mathrm{REnt}(X\|Y) = \sum_{x \in \mathcal{X}} \Pr\{X = x\} \log_2 \frac{\Pr\{X = x\}}{\Pr\{Y = x\}}.$$

В литературе энтропия X относительно Y называется также дискриминацией между X и Y, информационной дивергенцией X и Y, дивергенцией Кульбака — Лейблера X и Y, расстоянием Кульбака. Заметим, что значение $\mathrm{REnt}(X\|Y)$ конечно тогда и только тогда, когда $\mathcal{X} \subseteq \mathcal{Y}$. В частности, если $\Pr\{X = x\}$ мала (но отлична от 0) и $\Pr\{Y = x\} = 0$ для некоторого $x \in X$, то $\mathrm{REnt}(X\|Y) = +\infty$, что не вполне соответствует интуитивному представлению о близости распределений. Определим *энтропию X относительно Y при условии Z* следующей формулой:

$$\mathrm{REnt}((X|Z)\|(Y|Z)) = \sum_{z \in \mathcal{Z}} \Pr\{Z = z\}\ \mathrm{REnt}((X|Z = z)\|(Y|Z = z)).$$

Здесь через $\mathrm{REnt}((X|Z = z)\|(Y|Z = z))$ обозначена энтропия X относительно Y при условии $Z = z$, определение которой можно получить, заменив в приведенной выше формуле для $\mathrm{REnt}(X\|Y)$ все вероятности на условные, при условии $Z = z$. Известно, что $\mathrm{REnt}(X\|Y) \geqslant 0$, причем $\mathrm{REnt}(X\|Y) = 0$ тогда и только тогда, когда X и Y распределены одинаково. Поэтому $\mathrm{REnt}((X|Z)\|(Y|Z)) \geqslant 0$, причем $\mathrm{REnt}((X|Z)\|(Y|Z)) = 0$ тогда и только тогда, когда $(X|Z = z)$ и $(Y|Z = z)$ распределены одинаково для любого $z \in \mathcal{Z}$.

Пусть ε — произвольное вещественное неотрицательное число.

Определение 1 ([4]). Стеганосистема (Emb, Ext) называется

- *ε-стойкой по Кашену*, если $\mathrm{REnt}(C'\|S) \leqslant \varepsilon$;

- *абсолютно стойкой по Кашену*, если она является 0-стойкой по Кашену, т. е. если случайные величины C' и S распределены одинаково.

В этом определении речь идет о стойкости стеганосистемы против угрозы обнаружения стеганографического канала на основе атаки с известным стего. Определение допускает следующее очевидное обобщение на случай атаки с известным контейнером.

Определение 2. Стеганосистема (Emb, Ext) называется

- *ε-стойкой (по Кашену) против угрозы обнаружения стеганографического канала на основе атаки с известным контейнером*, если $\mathrm{REnt}((C'|C)\|(S|C)) \leqslant \varepsilon$.

Определение 3 ([15]). Стеганосистема (Emb, Ext) называется

- *ε-стойкой по Цёллнеру и др.*, если $\mathrm{Ent}(M) - \mathrm{Ent}(M|C,\, S) \leqslant \varepsilon$;
- *абсолютно стойкой по Цёллнеру и др.*, если она является 0-стойкой по Цёллнеру и др., т. е. если случайные величины M и $(C,\, S)$ независимы.

4.3. Результаты

Пусть стеганосистема (Emb, Ext) является ε-стойкой по Кашену. Тогда для оценки средних вероятностей α и β ошибок первого и второго рода Кашен [4] предлагает следующий хорошо известный в математической статистике метод. Известно, что для любой функции f на множестве $\mathcal{X} \cup \mathcal{Y}$ справедливо неравенство.

$$\mathrm{REnt}(f(X)\|f(Y)) \leqslant \mathrm{REnt}(X\|Y). \tag{1}$$

Для произвольных вещественных ξ и η таких, что $0 \leqslant \xi,\, \eta \leqslant 1$, положим

$$\delta(\xi,\, \eta) = \begin{cases} \xi\, \log_2(\xi/(1-\eta)) + (1-\xi)\, \log_2((1-\xi)/\eta), & \text{если } \xi \notin \{0,\, 1\}; \\ \log_2(1/\eta), & \text{если } \xi = 0; \\ \log_2(1/(1-\eta)), & \text{если } \xi = 1. \end{cases}$$

Тогда из неравенства (1), неравенства Йенсена и определения ε-стойкости по Кашену вытекает следующая теорема.

Теорема 1 ([4]). *Пусть* (Emb, Ext) — *стеганосистема*, *ε-стойкая по Кашену. Тогда*

$$\delta(\alpha,\, \beta) \leqslant \varepsilon.$$

В частности, если $\alpha = 0$, то $\beta \geqslant 2^{-\varepsilon}$.

Очевидно, что утверждение этой теоремы справедливо и для стеганосистемы, ε-стойкой против угрозы обнаружения стеганографического канала на основе атаки с известным контейнером (в смысле определения 2).

В следующей теореме рассматривается детерминированный случай, когда C' совпадает с исходным контейнером, т. е. когда функция G тождественна.

Теорема 2 ([15]). *Пусть* (Emb, Ext) — *стеганосистема*, *абсолютно стойкая по Цёллнеру и др. Тогда*

$$\mathrm{Ent}(S|C) = \mathrm{Ent}(C|S) = 0.$$

Этот результат показывает, что в детерминированном случае абсолютная стойкость невозможна, и теоретически обосновывает необходимость недетерминированности функции G.

5. Теоретико-сложностной подход

Как известно, в математической криптографии всякое определение стойкости криптографического протокола может быть сформулировано в «алгоритмическом стиле». Отсутствие метода взлома криптографического протокола формализуется посредством требования несуществования алгоритма, решающего стоящую перед противником задачу. Если на алгоритмы, используемые противником, не накладывается никаких ограничений, то говорят об абсолютной (синонимы — шенноновская, теоретико-информационная) стойкости. Формализации, получаемые при ограничении вычислительных ресурсов противника, дают определение теоретико-сложностной стойкости.

Ситуация в математической стеганографии, в целом, аналогична. Существуют теоретико-информационный подход, обсуждавшийся в разделе 4, и теоретико-сложностной подход, которому посвящен настоящий раздел. Не следует, однако, забывать, что, как уже отмечалось во введении, внешняя среда, в которой должны функционировать стеганосистемы, имеет большое количество степеней свободы. Поэтому абсолютная стойкость стеганосистемы на самом деле весьма относительна. В частности, в большинстве случаев источником контейнеров служит достаточно сложный и не контролируемый отправителем (Алисой) физический процесс. При разработке стеганосистемы используется математическая модель этого источника. И если стеганосистема является абсолютно стойкой относительно модельного источника контейнеров, то при переходе к реальному источнику эта стойкость может быть потеряна.

Такого же рода проблема возникает и в случае теоретико-сложностного подхода. Но есть и существенное различие. В самом деле, если противник создал, имея доступ к источнику контейнеров и используя только эффективные алгоритмы, достаточно точную модель

этого источника, то почему бы разработчику стеганосистемы не сделать то же самое?

5.1. Модель стеганосистемы

Всюду ниже $\mathbb{N}$ обозначает множество натуральных чисел, $n \in \mathbb{N}$ — параметр безопасности. Функции из $\mathbb{N}$ в $\mathbb{N}$, имеющие порядок роста n^γ, где $\gamma = \text{const}$, $\gamma > 0$, мы будем называть полиномиально растущими.

Пусть $C = \{C_n\}$, где $C_n \subseteq \{0,1\}^q$, — множество всех исходных контейнеров. Здесь $q = q(n)$ — полиномиально растущая функция. На множествах C_n не задается никакого распределения вероятностей. Не требуется также, чтобы эти множества были в каком-либо смысле плотными в соответствующих множествах $\{0,1\}^q$. В частности, можно считать, что для всякого n множество C_n содержит в точности один контейнер.

Определение 4. Последовательность контейнеров $\{c_n\}$, $c_n \in C_n$, называется полиномиально генерируемой, если существует (детерминированный) алгоритм, который на входе 1^n работает за полиномиальное (от n) время и выдает c_n.

Предполагается, что каждый исходный контейнер перед передачей по каналу связи подвергается некоторым случайным искажениям. Допустимые искажения задаются алгоритмически. Формально, пусть G — полиномиальная вероятностная машина Тьюринга, которая для всякого $c \in C_n$ вычисляет $c' = G(c)$, $c' \in \{0,1\}^q$. Здесь c' — случайная величина. Для дальнейшего нам удобнее рассматривать этот алгоритм G как детерминированный. Такой переход осуществляется с помощью следующего стандартного приема.

Поскольку машина G работает за полиномиальное время, существует полиномиально растущая функция $l\colon \mathbb{N} \to \mathbb{N}$ такая, что на любом входном слове длины n машина G использует не более $l(n)$ случайных битов. Пусть $c \in C_n$ и $r \in \{0,1\}^{l(n)}$. Тогда $c' = G(c, r)$. Иными словами, мы можем рассматривать машину G как детерминированную, интерпретируя содержимое ее случайной ленты как дополнительное входное слово. В дальнейшем мы будем называть алгоритм G генератором допустимых контейнеров. Этот алгоритм считается общеизвестным.

Мы рассматриваем модель стеганографического канала без повторений. Противник в нашей модели пассивный и полиномиально ограниченный. Это означает, что в его распоряжении имеется полиномиальная вероятностная машина Тьюринга A, которая на входе $s \in \{0,1\}^q$ выдает либо 0, либо 1. Если $A(s) = 1$, то контейнер s считается допустимым, т. е. Уэнди пропускает его по каналу связи между Алисой и Бобом. Если $A(s) = 0$, то контейнер s считается подозрительным и не пропускается.

Пусть $\alpha\colon \mathbb{N} \to (0,1]$. Основное предположение о действиях и возможностях противника следующее. Пусть $c \in C_n$, $r \in_R \{0,1\}^l$ и $c' = G(c, r)$. Тогда

$$\Pr\{A(c') = 1\} \geqslant \alpha(n).$$

Вероятность здесь определяется случайным выбором строки r и случайными величинами, которые использует в своей работе машина A. Интуитивно, $\alpha(n)$ — это порог, ограничивающий «злоумышленность» противника. Заметим, что в работах по стеганографии обычно предполагается, что используется «абсолютно надежный» источник контейнеров, для которого $\alpha \equiv 1$.

Стойкость стеганосистемы определяется относительно пары (атака с известным стего, угроза обнаружения стеганографического канала). Это означает, что противнику известна стеганосистема, за исключением ключа, а также контейнер s, передаваемый по каналу связи. Угроза состоит в обнаружении стеганографического канала, т. е. задача противника — установить, что представляет собой строка s — пустой контейнер или стего.

Определение 5 ([1]). Стеганосистема (Emb, Ext) для данного генератора допустимых контейнеров G называется стойкой относительно пары (атака с известным стего, угроза обнаружения стеганографического канала), если для любой полиномиально генерируемой последовательности контейнеров $\{c_n\}$, для любой последовательности сообщений $\{m_n\}$, $m_n \in \{0,1\}^{l(n)}$, семейства распределений вероятностей $\{G(c_n)\}$ и $\{\mathrm{Emb}(c_n, m_n, k)\}$, где $k \in_R \{0,1\}^n$, полиномиально неразличимы.

Напомним, что два семейства распределений вероятностей $\{D_n\}$ и $\{D'_n\}$ называются полиномиально неразличимыми, если для любой полиномиальной вероятностной машины Тьюринга B, для любого полинома p и для всех достаточно больших n

$$\left| \Pr_{x \in D_n}\{B(x) = 1\} - \Pr_{x \in D'_n}\{B(x) = 1\} \right| < 1/p(n).$$

В нашем случае распределения D_n и D'_n заданы на двоичных строках длины $q(n)$.

Замечание 1. Требование полиномиальной генерируемости последовательности контейнеров $\{c_n\}$ можно заменить, например, требованием общедоступности исходных контейнеров и формализовать его посредством оракула, к которому могут обращаться все рассматриваемые алгоритмы. Тогда приводимые ниже результаты будет верны в универсуме, релятивизированном к этому оракулу.

Очевидно, что стойкость стеганосистемы в смысле определения 5 слабее теоретико-информационной стойкости и является тривиальным

следствием последней. Для криптосистем, обладающих теоретико-информационной стойкостью, хорошо известна нижняя оценка Шеннона: длина ключа должна быть не меньше длины открытого теста. Аналогичные оценки доказаны и для стеганосистем, обладающих теоретико-информационной стойкостью (см., например, [15]). Нас же интересует случай $t(n) \geqslant n+1$. Всюду ниже предполагается, что это неравенство выполняется для всех (достаточно больших) n.

5.2. Необходимое условие существования стойких стеганосистем

Теорема 3 ([1]). *Если существует стеганосистема с детерминированным алгоритмом* Emb, *стойкая в смысле определения* 5, *то существуют односторонние функции.*

Для доказательства теоремы 3 нам потребуются следующие два определения.

Определение 6. Пусть $\{D_n\}$ — семейство распределений вероятностей, где D_n определено на $\{0,1\}^{\beta(n)}$. Семейство $\{D_n\}$ называется полиномиально конструируемым (polynomially samplable), если существует полиномиальная вероятностная машина Тьюринга Samp такая, что $\mathrm{Samp}(1^n) = D_n$.

Определение 7 ([10]). Пусть $g\colon \{0,1\}^n \to \{0,1\}^{\beta(n)}$ — функция, вычислимая за полиномиальное время. Функция g называется генератором дополнительной энтропии, если существуют полином p и полиномиально конструируемое семейство распределений $\{D_n\}$ такие, что

1. Семейства распределений $\{D_n\}$ и $\{g(x)\}$, $x \in_R \{0,1\}^n$, полиномиально неразличимы.
2. $\mathrm{Ent}(D_n) \geqslant \mathrm{Ent}(g(x)) + 1/p(n)$.

Здесь $\mathrm{Ent}(Y)$ обозначает шенноновскую энтропию случайной величины Y.

Доказательство теоремы 3. Общая схема доказательства такова. В предположении существования стеганосистемы, стойкой в смысле определения 5, доказывается, что существуют генераторы дополнительной энтропии. В работе Импальяццо, Левина и Луби [10] (см. также [12]) доказано, что если существуют генераторы дополнительной энтропии, то существуют и псевдослучайные генераторы. Хорошо известно [10], что существование последних эквивалентно существованию односторонних функций.

Пусть существует стеганосистема, стойкая в смысле определения 5, и пусть G — соответствующий генератор допустимых контейнеров.

Генератор дополнительной энтропии g строится следующим образом. Пусть $\{c_n\}$ — полиномиально генерируемая последовательность контейнеров. Зафиксируем какой-либо полиномиальный (детерминированный) алгоритм, который на входе 1^n выдает сообщение $m_n \in \{0,1\}^{t(n)}$. Пусть $x \in \{0,1\}^n$ — входное слово генератора g. Положим $g(x) = \mathrm{Emb}(c_n, m_n, x)$. Очевидно, что функция g вычислима за полиномиальное время. Очевидно также, что $\mathrm{Ent}(g(x)) \leqslant n$.

Далее, согласно определению 5 семейства распределений $\{g(x)\}$, $x \in_R \{0,1\}^n$, и $\{G(c_n)\}$ полиномиально неразличимы. Напомним, что каждое распределение вероятностей из семейства $\{G(c_n)\}$ определяется равновероятным выбором строки r из $\{0,1\}^l$. Легко видеть, что семейства распределений $\{G(c_n)\}$ и $\{\mathrm{Emb}(c_n, m_n, k)\}$, где $k \in_R \{0,1\}^n$, а $m_n \in_R \{0,1\}^{t(n)}$, также полиномиально неразличимы. Из этого следует полиномиальная неразличимость семейств $\{g(x)\}$, $x \in_R \{0,1\}^n$, и $\{\mathrm{Emb}(c_n, m_n, k)\}$, $k \in_R \{0,1\}^n$, $m_n \in_R \{0,1\}^{t(n)}$. Ясно также, что семейство $\{\mathrm{Emb}(c_n, m_n, k)\}$, $k \in_R \{0,1\}^n$, $m_n \in_R \{0,1\}^{t(n)}$, полиномиально конструируемо и его энтропия не меньше $t(n)$ (это следует из однозначности извлечения скрытого сообщения m_n из стего $\mathrm{Emb}(c_n, m_n, k)$ алгоритмом Ext). По предположению $t(n) \geqslant n+1$, что доставляет требуемое неравенство для энтропий. □

Замечание 2. Можно предположить, что существование односторонних функций является необходимым условием для существования стойких стеганосистем и в том случае, когда алгоритм Emb вероятностный. Вопрос о том, так ли это на самом деле, остается открытым.

На первый взгляд может показаться, что существует тривиальное доказательство теоремы 3 посредством следующего рассуждения: для стойких стеганосистем функция, вычисляемая алгоритмом Emb, должна быть односторонней. В противном случае алгоритм ее инвертирования будет по заданному стего с достаточно большой вероятностью находить пары (ключ, сообщение), что, казалось бы, можно использовать, чтобы отличать стего от пустых контейнеров. Однако, из условия теоремы и определения 5 следует лишь существование генератора допустимых контейнеров G, для которого имеется стойкая стеганосистема. О распределении вероятностей, создаваемом алгоритмом G на множестве контейнеров, ничего не известно. Поэтому неясно, как будет вести себя алгоритм инвертирования на входах, выбранных из этого распределения. Более того, в случае стеганосистемы для скрытого канала носители распределений $G(c_n)$ и $\mathrm{Emb}(c_n, m_n, k)$, $k \in_R \{0,1\}^n$, совпадают для всякого n.

Вопрос о том, является ли существование односторонних функций достаточным условием для существования стеганосистем, стойких в смысле определения 5, остается открытым. Несложно доказать, что

это условие и в самом деле достаточно в случае, когда генератор допустимых контейнеров является частично обратимым.

Определение 8. Пусть G — генератор допустимых контейнеров и пусть для всякой строки $r \in \{0,1\}^l$, $r = r_1 \| r_2$, где $r_1 \in \{0,1\}^{l_1(n)}$, $l_1(n)$ — полиномиально растущая функция из $\mathbb{N}$ в $\mathbb{N}$ такая, что $l_1(n) < l(n)$. Генератор G называется частично обратимым, если существует полиномиальный алгоритм Der такой, что для всякого c и всякой строки $r \in \{0,1\}^l$, $\mathrm{Der}(G(c, r)) = r_1$.

Заметим, что хотя данный сценарий с теоретической точки зрения не слишком интересен, именно для него разработано большинство описанных в литературе конкретных стеганосистем.

Всюду в дальнейшем будем предполагать, что $l_1(n) \geqslant t(n)$.

В данном случае мы рассматриваем стойкость стеганосистемы относительно пары (атака с выбором контейнера и с выбором скрытого сообщения, угроза обнаружения стеганографического канала). Под противником понимается полиномиальная вероятностная машина Тьюринга A, удовлетворяющая сформулированному выше требованию (ограничивающему ее поведение на входах, созданных генератором допустимых контейнеров). Атака на стеганосистему состоит из двух фаз. На первой машина A получает на вход 1^n и выдает $m_n \in \{0,1\}^{t(n)}$, $c_n \in \{0,1\}^{q(n)}$. Разумеется, в этом случае m_n и c_n — случайные величины.

На второй фазе машина A получает на вход строку $s \in \{0,1\}^q$ и выдает либо 0, либо 1. Пусть $P_1(n) = \Pr\{A(s) = 1\}$ для случая, когда $s = G(c_n, r)$. Эта вероятность определяется выбором случайной строки r и случайными величинами самой машины A. Пусть $P_2(n) = \Pr\{A(s) = 1\}$ в случае, когда $s = \mathrm{Emb}(c_n, m_n, k)$. Вероятность P_2 определяется случайным выбором секретного ключа k и случайной строкой машины A.

Предполагается, что противник знает стеганосистему, за исключением ключа, и, кроме того, ему известны алгоритмы G и Der.

Определение 9. Стеганосистема (Emb, Ext) для данного генератора допустимых контейнеров G называется стойкой относительно пары (атака с выбором контейнера и с выбором скрытого сообщения, угроза обнаружения стеганографического канала), если для любой полиномиальной вероятностной машины Тьюринга A указанного выше типа, для любого полинома p и для всех достаточно больших n

$$|P_1(n) - P_2(n)| < 1/p(n).$$

Утверждение следующей теоремы носит в определенном смысле условный характер, поскольку оно предполагает существование частично обратимых генераторов допустимых контейнеров. Очевидно, что частично обратимый генератор можно построить без привлечения каких-

либо недоказанных гипотез. Но для того, чтобы создаваемые им контейнеры были допустимыми, требуется соответствующее предположение о действиях и возможностях противника.

Теорема 4 ([1]). *Если существуют односторонние функции, то существуют стеганосистемы, стойкие относительно пары (атака с выбором контейнера и с выбором скрытого сообщения, угроза обнаружения стеганографического канала).*

Доказательство. Пусть g: $\{0,1\}^n \to \{0,1\}^{t(n)}$ — псевдослучайный генератор. Его существование следует из существования односторонней функции [10], [12]. Стеганосистема (Emb, Ext) строится следующим образом. Алгоритм Emb на входе (c_n, m_n, k) сначала вычисляет $\sigma = g(k) \oplus m_n$. Затем он вызывает алгоритм G, подавая ему на вход c_n и $r = \sigma \| r'$, где $r' \in_R \{0,1\}^{l(n)-t(n)}$. Пусть $s = G(c_n, r)$. Строка s (стего) и является выходом алгоритма Emb.

Алгоритм Ext сначала вызывает алгоритм Der, подавая ему на вход строку s. Пусть σ' — префикс длины $t(n)$ строки $\mathrm{Der}(s)$. Тогда $m_n = \sigma' \oplus g(k)$.

Доказательство стойкости построенной стеганосистемы вытекает непосредственно из определения псевдослучайного генератора: никакой полиномиальный вероятностный алгоритм не может отличить псевдослучайную строку $g(k)$, $k \in_R \{0,1\}^n$, от случайной строки той же длины. □

5.3. О модели с активным противником

В некоторых работах по стеганографии утверждается, что задача создания стеганографического канала в присутствии активного противника является очень сложной и требует разработки методов, отличных от тех, которые позволяют защищаться от пассивного противника. Главный итог подобных рассуждений — весьма пессимистический прогноз, поскольку на данный момент, по существу, никаких специальных методов защиты от активного противника не предложено. Тем не менее, в данном подразделе мы выделяем один частный (но весьма важный с практической точки зрения) случай, в котором от активного противника можно защититься теми же методами, что и от пассивного.

Как правило, рассуждения об активном противнике укладываются в такую схему: поскольку скрытое сообщение «упрятывается» отправителем в некоторый псевдошум, добавляемый к контейнеру, активный противник всегда может это сообщение разрушить путем модификации псевдошума. Но все это справедливо лишь в том случае, когда при передаче сообщений между Алисой и Бобом возникает шум в канале.

Если же сообщения (например, файлы) передаются через компьютерную сеть, то они доставляются получателю практически без искажений. В данном подразделе мы рассматриваем именно этот случай бесшумного канала.

Далее, в классической задаче о двух заключенных предполагается, что Уэнди осуществляет цензурирование сообщений, пересылаемых между Алисой и Бобом, на законных основаниях. На практике, однако, такая ситуация встречается крайне редко. Поэтому мы будем предполагать, что активный противник должен скрывать свои действия от отправителя и получателя.

Итак, рассматривается следующая модель. Пусть, как и прежде, $c \in C_n$ — исходный контейнер, известный Алисе. С помощью алгоритма G она создает пустой контейнер $c' = G(c)$. Напомним, что m обозначает скрытое сообщение, (Emb, Ext) — стеганосистему, а k — ее секретный ключ. Алиса передает Бобу либо пустой контейнер c', либо стего $s = \mathrm{Emb}(c, m, k)$. Уэнди перехватывает передаваемый контейнер $\tilde{c}$ и с помощью полиномиального вероятностного алгоритма подмены Sub создает контейнер $c'' = \mathrm{Sub}(\tilde{c})$, который и передается Бобу. При этом действия алгоритма подмены должны быть незаметны для Боба, что формализуется требованием полиномиальной неразличимости семейств распределений $\{G(c)\}$ и $\{\mathrm{Sub}(c')\}$, $c' = G(c)$. Подчеркнем, что это условие никак не ограничивает действия Уэнди в том случае, когда по каналу связи пересылается стего.

Описанная выше модель подсказывает следующую методику использования стеганосистемы. Алиса должна послать Бобу скрытое сообщение m. Прежде чем отправить соответствующее стего s, Алиса выбирает некоторое количество исходных контейнеров $c_1, \ldots, c_u$, создает с помощью алгоритма G соответствующие пустые контейнеры $c'_1, \ldots, c'_u$, которые мы будем называть пробными, и посылает их Бобу. Назначение этой предварительной фазы — обнаружить присутствие в канале активного противника. Если на каком-либо из пробных контейнеров Боб обнаружит вмешательство Уэнди, то он подает Алисе условный знак (любым возможным способом) и стего не передается.

Разумеется, задача становится тривиальной, если Боб знает в точности все передаваемые пробные контейнеры $c'_1, \ldots, c'_u$ или может их отправить обратно Алисе (минуя Уэнди!) на проверку.

Параметр u считается неизвестным Уэнди и, вообще говоря, Бобу и может выбираться, например, случайным образом из множества $V = \{1, \ldots, v\}$. Мы считаем, что Уэнди не знает значение v. Это соответствует следующей реальной ситуации: по каналу связи между Алисой и Бобом передается некоторый поток контейнеров, среди которых могут быть как пустые, так и стего. Уэнди не имеет никакой априорной информации, которая позволяла бы разбивать этот поток на подпоследовательности связанных между собой контейнеров.

Определение 10. Стеганосистема (Emb, Ext) для данного генератора допустимых контейнеров G называется ε-стойкой против активного противника (против угрозы разрушения скрытого сообщения на основе атаки с известным стего), если:

1) она является стойкой против пассивного противника (см. определение 5);
2) для любой полиномиальной вероятностной машины Тьюринга Sub справедливо следующее. Пусть s' — контейнер, полученный Бобом, когда Алиса передавала стего s, т. е. либо $s' = s$, либо $s' = \mathrm{Sub}(s)$. Тогда $\Pr\{\text{Алиса передала } s \ \&\ \mathrm{Ext}(s') \neq m\} \leqslant \varepsilon$.

Фактически, п. 10 определения означает, что с помощью пробных контейнеров Алиса и Боб либо обнаруживают присутствие в канале активного противника, либо убеждаются, что канал достаточно безопасен и Боб с высокой вероятностью получит скрытое сообщение m.

Общая идея построения стеганосистемы, которая, предположительно, была бы стойкой в смысле определения 10, достаточно очевидна. Алиса выбирает $u \in_R V$ и создает пробные контейнеры $c'_1, \ldots, c'_u$, которые содержат заранее условленные метки. Если Боб обнаружит, что хотя бы одна из этих меток изменена, то он выдает «сигнал тревоги». В противном случае, т. е. когда такого сигнала не поступило, в качестве $(u+1)$-го контейнера Алиса передает стего.

Если G — частично обратимый генератор допустимых контейнеров, то описанную выше идею можно реализовать следующим образом. Секретным ключом стеганосистемы является пара (k_1, k_2), где $k_1, k_2 \in \{0, 1\}^n$. Пусть $g\colon \{0, 1\}^n \to \{0, 1\}^{v \cdot l_1(n)}$ — псевдослучайный генератор. Напомним, что l_1 — длина той части случайной строки, используемой генератором допустимых контейнеров G, которая может быть извлечена из контейнера с помощью алгоритма Der (см. подраздел 5.2). Алиса вычисляет $g(k_1)$ и делит полученную последовательность на v блоков $r_1, \ldots, r_v$, каждый длины $l_1(n)$. Затем она выбирает исходные контейнеры $c_1, \ldots, c_u$, вычисляет пробные контейнеры $c'_1 = G(c_1, r_1), \ldots, c'_u = G(c_u, r_u)$ и пересылает их Бобу. Последний с помощью алгоритма Der извлекает из пробных контейнеров строки $r_1, \ldots, r_u$ и сравнивает их с соответствующими блоками последовательности $g(k_1)$. Ясно, что любое вмешательство Уэнди, которое привело к изменению хотя бы одного из этих l_1 битов хотя бы в одном пробном контейнере, будет обнаружено. В противном случае, Алиса, используя секретный ключ k_2 создает, как это описано в предыдущем подразделе,

стего s, содержащее скрытое сообщение m, и посылает s Бобу. Очевидно, что для разрушения скрытого сообщения Уэнди должна либо угадать значение u, либо уметь отличать стего от пустых контейнеров.

5.4. Случай неравномерно распределенного шума

Выше мы рассматривали модель стеганосистемы в предположении существования частично обратимого генератора допустимых контейнеров. Нетрудно видеть, что это предположение на самом деле содержит в себе две гипотезы:

- существует эффективный алгоритм (частичного) извлечения шума из контейнера;
- шум, извлеченный из контейнера, распределен равномерно.

Как с практической, так и с теоретической точек зрения представляет интерес проблема построения стойких стеганосистем в случае, когда выполняется только первая из этих гипотез, т. е. шум имеет распределение, отличное от равномерного. В данном подразделе мы рассматриваем эту проблему при следующих предположениях:

- Алиса имеет доступ к источнику контейнеров как к оракулу;
- контейнеры сами являются «шумом», т. е. на каждый запрос оракул возвращает битовую строку, выбранную из распределения, близкого к равномерному;
- параметры распределения вероятностей на множестве контейнеров Алисе не известны. Она знает лишь, что это распределение близко к равномерному;
- рассматривается модель стеганографического канала без повторений;
- все контейнеры, полученные Алисой от оракула, пересылаются по каналу связи Бобу. Для передачи скрытого сообщения Алиса может выбрать любой из этих контейнеров. Все вопросы о том, каким образом Боб сможет понять, какой из контейнеров является стего, чтобы корректно извлечь из него скрытое сообщение, игнорируются.

Таким образом, рассматриваемая модель основывается на весьма сильных предположениях в пользу отправителя (или, точнее, в пользу разработчика стеганосистемы). Но поскольку в итоге будет получен отрицательный результат, такие предположения этот результат лишь усиливают.

Пусть n — параметр безопасности, $u = u(n)$ — полиномиально растущая функция. Алиса получает от оракула контейнеры $c_1, \ldots, c_u$, каждый длины n, выбирает $i \in \{1, \ldots, u\}$ и заменяет контейнер c_i на стего

$s = \mathrm{Emb}(c_i, m, k)$, где k — секретный ключ стеганосистемы, а m — скрытое сообщение. Можно сделать еще одно предположение в пользу отправителя и считать скрытое сообщение m случайной, относительно равномерного распределения, строкой длины $t < n$.

В данной модели стойкость стеганосистемы относительно пары (атака с известным стего, угроза обнаружения стеганографического канала) определяется требованием вычислительной неразличимости семейств распределений $\{c_1, \ldots, c_{i-1}, c_i, c_{i+1}, \ldots, c_u\}$ и $\{c_1, \ldots, c_{i-1}, s, c_{i+1}, \ldots, c_u\}$. Для всякого значения параметра n распределение вероятностей из первого семейства определяется случайным выбором контейнеров, а распределение вероятностей из второго семейства — еще и случайным выбором индекса i, ключа k, скрытого сообщения m, а также случайными величинами алгоритма Emb. Заметим, что выбор индекса i не обязательно должен быть случайным и может рассматриваться как часть алгоритма Emb.

Мы не приводим формального определения стойкости, поскольку оно является очевидной модификацией определения 5.

Определение 11. Пусть D — распределение вероятностей на множестве $\{0, 1\}^n$ и пусть X — случайная величина, принимающая значения в $\{0, 1\}^n$ в соответствии с распределением D. Тогда min-энтропией случайной величины X называется величина

$$\mathrm{Ent}_{\min}(X) = \min_{x \in \{0,1\}^n} (-\log \Pr\{X = x\}).$$

Вместо min-энтропии можно рассматривать близкое ей понятие энтропии Реньи.

Определение 12. Пусть X и Y — независимые одинаково распределенные случайные величины. Энтропией Реньи случайной величины X называется величина

$$\mathrm{Ent}_{\mathrm{Ren}}(X) = -\log \Pr\{X = Y\}.$$

Из теории вероятностей известно, что min-энтропия и энтропия Реньи связаны между собой следующими неравенствами

$$\mathrm{Ent}_{\mathrm{Ren}}(X)/2 \leqslant \mathrm{Ent}_{\min}(X) \leqslant \mathrm{Ent}_{\mathrm{Ren}}(X).$$

То есть эти энтропии совпадают с точностью до мультипликативной константы 2. Кроме того, нетрудно показать, что энтропия Реньи всегда не превосходит шенноновскую энтропию.

Определение 13. Распределение $X = (X_1, \ldots, X_u)$ над множеством $\{0, 1\}^{nu}$ называется (n, h)-блоковым источником, если для всех $i = 1, \ldots, u$ и для каждого $z \in \{0, 1\}^{nu-n}$

$$\mathrm{Ent}_{\min}(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_u = z) \geqslant h.$$

Здесь $\mathrm{Ent}_{\min}(X_i|X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_u = z)$ — условная min-энтропия случайной величины X_i при условии $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_u = z$.

Под источником энтропии понимается $(n, n - 1/p(n))$-блоковый источник, где p — некоторый фиксированный полином. Заметим, что (n, n)-блоковый источник соответствует равномерному распределению вероятностей на множестве всех битовых строк длины nu. Так что строки, выбираемые из $(n, n - 1/p(n))$-блокового источника, весьма незначительно отличаются от чисто случайных строк.

Будем говорить, что стеганосистема описанного выше вида является *тривиальной*, если вероятность, что $s \neq c_i$, пренебрежимо мала как функция от n.

Теорема 5. *Если для данного фиксированного полинома p существует стеганосистема, стойкая для любого $(n, n - 1/p(n))$-блокового источника контейнеров, то эта стеганосистема является тривиальной.*

Эта теорема является простым следствием следующей леммы из работы [9]. В формулировке леммы Γ — класс всех $(n, n - 1/p(n))$-блоковых источников с u блоками, $N = nu$. Значения параметров u, N' и M ограничены сверху полиномами от n.

Лемма 1. *Пусть функции F: $\{0, 1\}^N \times \{0, 1\}^{N'} \to \{0, 1\}^M$ и G: $\{0, 1\}^N \to \{0, 1\}^M$ и распределение вероятностей Y на множестве $\{0, 1\}^{N'}$ таковы, что для любого распределения вероятностей $X \in \Gamma$ семейства распределений $\{F(X, Y)\}$ и $\{G(X)\}$ полиномиально неразличимы. Тогда вероятность $\Pr\{F(x, y) \neq G(x)\}$ пренебрежимо мала как функция от n. Здесь $x \in_R \{0, 1\}^N$, а y выбирается из множества $\{0, 1\}^{N'}$ в соответствии с распределением Y.*

Замечание 3. Из данной леммы в работе [9] выведен ряд результатов о несуществовании различных криптографических протоколов и примитивов (шифрование, разделение секрета, доказательства с нулевым разглашением и т. п.) в модели со слабыми источниками случайности (т. е. в модели, где источник чисто случайных битов заменен источником энтропии). При всем сходстве этих результатов с теоремой 5 имеется и принципиальное отличие. Выбор источника случайности для криптографических конструкций находится в руках разработчика, так что отрицательные результаты лишь указывают на необходимость проведения дополнительных исследований с целью отбора подходящих источников. А для стеганосистемы источник контейнеров представляет собой часть исходных данных.

5.5. Стеганография с «черным ящиком»

В работе Дедича и др. [8] исследуется возможность построения стойких стеганосистем в случае, когда источником контейнеров является оракул, или, иначе говоря, «черный ящик». Отправитель может выдавать запросы оракулу и в ответ получать пустые контейнеры, выбираемые из некоторого источника энтропии. Существенное отличие от модели, рассматривавшейся в предыдущем подразделе состоит в том, что отправитель имеет возможность принимать решение, что делать с полученным от оракула контейнером: преобразовать в стего и отослать получателю, или игнорировать.

Данная модель интересна тем, что позволяет уйти от далекого от реальности предположения, согласно которому отправителю (а, значит, и разработчику стеганосистемы) известны все требуемые параметры источника контейнеров, в т. ч. распределение вероятностей на множестве пустых контейнеров.

Рассматривается стеганографический канал без повторений. Стойкость стеганосистемы против угрозы обнаружения стеганографического канала на основе атаки с выбором скрытого сообщения определяется в теоретико-сложностном стиле [8]. Исследуется среднее количество запросов к оракулу, выдаваемых стеганосистемой, как функция от min-энтропии источника контейнеров. Доказана экспоненциальная нижняя оценка количества запросов.

Переходим к формальному изложению основного результата работы [8].

Пусть Σ — конечный алфавит. Под каналом C понимается отображение, которое преобразует предысторию $H \in \Sigma^*$ в распределение вероятностей D_H на множестве Σ. Предыстория $H = s_1 s_2 \ldots s_l$ называется легальной, если всякий ее символ может быть получен, исходя из предшествующих символов, т. е. $\Pr_{D_{s_1 \ldots s_{i-1}}}(s_i) > 0$.

В стеганосистемах, рассматриваемых в работе [8], доступ отправителя к каналу C формализуется посредством оракула M, который на запрос H возвращает символ s, представляющий собой реализацию случайной величины с распределением вероятностей D_H.

Определение 14. Стеганосистемой в модели с черным ящиком называется пара $S = (\mathrm{Emb}, \mathrm{Ext})$ полиномиальных вероятностных алгоритмов таких, что:

1. Входными данными алгоритма Emb служат секретный ключ k, строка $m \in \{0, 1\}^*$ (скрытое сообщение), и предыстория H. Кроме того, этот алгоритм имеет доступ к оракулу M. Выходом алгоритма Emb является строка символов $s_1 s_2 \ldots s_l \in \Sigma^*$ (стеготекст).

2. Входными данными алгоритма Ext служат секретный ключ k, предыстория H и стеготекст $s_1 s_2 \ldots s_l \in \Sigma^*$. Алгоритм выдает скрытое сообщение $m \in \{0, 1\}^*$.

Здесь $k \in_R \{0, 1\}^n$, где n — параметр безопасности.

Авторы предполагают, что длина l стеготекста зависит только от длины скрытого сообщения m, но не от его содержания. Под скоростью стеганографического канала понимается число $w > 0$ такое, что для передачи скрытого сообщения длины lw требуется в среднем l символов стеготекста.

Надежность стеганосистемы. Надежность стеганосистемы S с параметром безопасности n, для канала C и сообщений длины l определяется как

$$\mathrm{Rel}_S = \min \Pr\{\mathrm{Ext}(H, \mathrm{Emb}^M(H, m, k), k) = m\}.$$

Минимум берется по всем легальным предысториям H и всем $m \in \{0,1\}^l$. Вероятность определяется случайным выбором ключа $k \in \{0, 1\}^n$ и случайными величинами алгоритмов Emb и Ext и оракула M.

Рассматривается пассивный противник, проводящий атаку с выбором скрытого сообщения. Цель такого противника — отличить стего, содержащее выбранное им (противником) скрытое сообщение от случайной выборки из канала C. Для формализации определяется оракул $O(H, \cdot)$, который на входе (H, m), где $m \in \{0, 1\}^*$, H — предыстория, действует следующим образом. Сначала O вычисляет длину l стеготекста, который выдал бы алгоритм Emb на входе (H, m, k), затем последовательно выбирает случайные s_i, каждое из распределения $D_{H\|s_1 \ldots s_{i-1}}$, и выдает $s_1 \ldots s_l$.

Определение 15. Вероятностный алгоритм W называется (t, d, q, l)-пассивным противником для стеганосистемы S, если:

1. W работает за время t в среднем (здесь учитывается время, необходимое алгоритму Emb для ответов на запросы противника) и имеет длину описания d.
2. W имеет доступ к каналу C, формализуемый посредством оракула M.
3. Второй оракул, к которому имеет доступ алгоритм W, — это либо $\mathrm{Emb}^M(\cdot, \cdot, k)$, либо $O(\cdot, \cdot)$. W может выдать этому оракулу в среднем q запросов суммарной длины l битов.
4. Выходом алгоритма W является один бит.

Стойкость стеганосистемы. Успешностью противника W против стеганосистемы S с параметром безопасности n и каналом C называется величина

$$\mathrm{Adv}_S(W) = \left|\Pr\left\{W^{M,\mathrm{Emb}^M(\cdot,\cdot,k)} = 1\right\} - \Pr\left\{W^{M,O(\cdot,\cdot)} = 1\right\}\right|.$$

Вероятности определяются случайными величинами алгоритма W и оракулов, а первая из вероятностей — еще и случайным выбором ключа k из множества $\{0, 1\}^n$.

Стойкость стеганосистемы для канала C определяется как функция

$$\mathrm{Sec}_S(t, d, q, l) = 1 - \max \mathrm{Adv}_S(W).$$

Максимум берется по всем (t, d, q, l)-пассивным противникам W.

Теорема 6. *Пусть S — стеганосистема со стойкостью $1 - \varepsilon$, надежностью $1 - \rho$ и скоростью w. Предположим, что противник имеет доступ к дополнительному оракулу, позволяющему проверять принадлежность символа s_i носителю распределения D_i. Тогда существует канал с min-энтропией h, для которого вероятность, что алгоритм* Emb *при отсылке случайного скрытого сообщения длины lw делает не более N запросов к оракулу M, не превосходит*

$$\left(\frac{Ne}{l2^w}\right)^l + \rho + \varepsilon R,$$

а, следовательно, среднее число запросов к оракулу на один символ стеготекста не меньше

$$\frac{2^w}{e}(1/2 - \rho - \varepsilon R),$$

где $R = 1/(1 - 2^h/|S|)$.

Этот результат, так же как и теорема 5, *демонстрирует невозможность построения стойких стеганосистем для источников энтропии (в качестве источников контейнеров). Подчеркнем, что речь идет об «универсальных» стеганосистемах, которые должны быть пригодны для любого источника контейнеров, энтропия которого достаточно велика. Эти отрицательные результаты не исключают возможности существования специализированной стеганосистемы для каждого отдельного источника энтропии.*

6. Электронные водяные знаки

Электронные водяные знаки и их специальная разновидность, электронные отпечатки пальцев, в последние годы стали весьма популярной темой исследований, привлекшей внимание специалистов из различных областей математики. Вместе с тем, этим исследованиям, на наш взгляд, не хватает самого главного, а именно, адекватной с точки зрения потенциальных приложений и математически корректной постановки задачи.

Начнем с замечания, что не существует единого мнения по поводу отнесения электронных водяных знаков к стеганографии. Как известно, стеганографические методы призваны скрывать сам факт передачи информации. В противоположность этому, наличие водяных знаков в какой-либо информации может быть общеизвестным и задача состоит в их защите от удаления и (или) изменения. Возможны два выхода из этой ситуации. Первый состоит в расширительном толковании термина «стеганография», включающем и электронные водяные знаки. Альтернативная точка зрения признает существование научной дисциплины под названием «сокрытие информации» (information hiding). Вся область исследований, связанных с проблематикой электронных водяных знаков, входит, наряду со стеганографией, в эту научную дисциплину.

Электронный водяной знак представляет собой некий аналог скрытого сообщения. Это — специальные данные, добавляемые в различного рода информацию, представленную в электронной форме, интеллектуальным собственником этой информации. Назначение электронного водяного знака — служить доказательством для третьих лиц, что данная информация является интеллектуальной собственностью того, кто внес в нее этот водяной знак. Электронные водяные знаки могут быть индивидуализированы. С этой целью продавец, поставляющий клиентам файл с какой-либо информацией, помещает в каждую копию этого файла водяной знак с данными, идентифицирующими клиента, которому данная копия была продана. В дальнейшем, в случае обнаружения пиратских копий файла, продавец сможет идентифицировать клиента, который эти копии распространяет.

Однако, индивидуализированные водяные знаки не позволяют продавцу доказывать третьим лицам, что пиратские копии распространяет именно данный клиент. Разновидность водяных знаков, обеспечивающая возможность такого доказательства, получила название электронных отпечатков пальцев. Доказательство может быть убедительным для третьих лиц только в том случае, если у каждого клиента имеется (сертифицированный) открытый ключ, а отпечаток пальцев содержит информацию, которую невозможно вычислить, не зная соответствующий секретный ключ. Здесь прослеживается аналогия с электронной подписью. Но отпечатки пальцев не являются подписями, поскольку могут быть «сняты», по крайней мере в принципе, без ведома клиента. Например, клиент может использовать свой секретный ключ для выполнения какого-либо протокола, предусмотренного процедурой купли-продажи. На основе транскрипции протокола продавец создает электронный отпечаток пальцев. При этом клиент может даже не подозревать, что полученная им копия файла содержит электронный отпечаток пальцев, и вообще не догадываться, что такие существуют в природе.

Проблемы, возникающие при попытке формализовать понятие электронного водяного знака, удобнее обсуждать на основе какого-либо определения, известного из литературы. Мы рассмотрим определение схемы электронных водяных знаков для защиты программного обеспечения из работы [6]. Под программой в этом определении понимается булева схема C (схема из функциональных элементов). Электронный водяной знак обозначается через m и выбирается из некоторого множества водяных знаков, имеющих длину, подходящую для программы C. Программа, содержащая электронный водяной знак, называется помеченной.

Используется также следующее соглашение. Для функции $\alpha\colon \{0,1\}^* \to [0, 1]$ формула $\forall\, x\ \alpha(x) = \nu(n)$ трактуется как сокращение следующего предложения: для любого полинома Q существует n_0: $\forall\, n \geqslant n_0$ $\forall\, x \in \{0, 1\}^n\ \alpha(x) < 1/Q(n)$.

Определение 16. Схемой электронных водяных знаков называется пара вероятностных алгоритмов (Mark, Extract) таких, что:

- (*функциональность*). Для всяких схемы C, ключа k и водяного знака m, строка $\mathrm{Mark}_k(C, m)$ описывает схему, которая вычисляет ту же функцию, что и C.
- (*эффективность помеченной программы*). Существует полином p такой, что для всякой схемы C, $|\mathrm{Mark}_k(C, m)| \leqslant p(|C| + |m| + |k|)$.
- (*извлекаемость*). Для всяких схемы C, ключа k, и водяного знака m, $\mathrm{Extract}_k(\mathrm{Mark}_k(C, m)) = m$.
- (*значимость*). Для всякой схемы C, $\Pr_k\{\mathrm{Extract}_k(C) \neq \lambda\} = \nu(|C|)$.
- (*хрупкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любых схемы C и водяного знака m

$$\Big|\Pr_k\{A(\mathrm{Mark}_k(C, m)) = C'\colon C' \approx C\ \&\ \mathrm{Extract}_k(C') \neq m\} - \Pr\big\{S^C(1^{|C|}) = C'\colon C' \approx C\big\}\Big| = \nu(|C|).$$

Ключ k выбран наудачу из множества $\{0, 1\}^{\max(|C|,|m|)}$, а $C' \approx C$ означает, что схемы C' и C функционально эквивалентны.

Напомним, что λ обозначает пустую строку.

Схема электронных водяных знаков называется эффективной, если функции Mark и Extract вычислимы за полиномиальное время.

В определении 16 свойство значимости отражает интуитивное требование, что большинство программ должны быть непомеченными (посредством водяных знаков). В самом деле, для любой программы C лишь ничтожная доля ключей k позволяет извлечь из программы что-либо отличное от пустой строки.

Из всех свойств, сформулированных в определении 16, основное внимание в литературе уделяется свойству хрупкости. В различных работах это свойство формулируется по-разному и даже неодинаково называется, но всегда формализует интуитивное требование стойкости: активный противник не может уничтожить водяной знак, не «разрушив» при этом весь файл. В том случае, когда электронные водяные знаки используются для защиты программного обеспечения, у противника всегда есть возможность проводить с программой эксперименты, получая пары (вход, выход). Если этой информации достаточно для того, чтобы построить программу, эквивалентную исходной, то водяные знаки как средство защиты не имеют смысла. Именно эта возможность учитывается в формулировке свойства хрупкости определения 16, где вероятность удаления противником водяного знака из помеченной программы $\mathrm{Mark}_k(C, M)$ сравнивается с вероятностью построения схемы, эквивалентной исходной схеме C, при доступе к последней как к оракулу.

И все же важнейшим свойством схемы электронных водяных знаков является извлекаемость. К сожалению, большинство авторов не уделяют этому свойству должного внимания. А ведь электронные водяные знаки представляют интерес только в том случае, если позволяют разрешать споры об интеллектуальной собственности. Предположим, что Боб торгует файлом F, а Алиса утверждает, что этот файл является ее интеллектуальной собственностью. Для разрешения спора она обращается к арбитру, предъявляя ему «полиномиальный вероятностный алгоритм» Extract и ключ k. Популярная теоретическая конструкция водяных знаков основывается на кодах, исправляющих ошибки, так что ключ k вполне может быть матрицей размера, скажем $10^5 \times 10^6$. Алгоритм Extract, проработав на входных данных F, k несколько часов (полиномиальное время!), выдаст водяной знак Алисы, содержащий ее Copyright. Насколько такое доказательство будет убедительным для арбитра? И что должен делать арбитр, если Боб предоставит свой ключ k', с помощью которого другой (или даже тот же самый) алгоритм Extract извлечет из файла F его (Боба) водяной знак?

Процедура извлечения водяного знака должна быть простой и понятной. Например, если файл F содержит графическую информацию, то

такая процедура, в принципе, могла бы выглядеть следующим образом. Алиса заявляет арбитру: «Замените все зеленые пиксели на черные, синие — на белые, и т. д., и посмотрите в левый нижний угол картинки. Вы увидите мой автограф.» Вероятно, такого рода доказательства могут быть убедительными для арбитра.

Ввиду того, что, на наш взгляд, не решена главная проблема на пути к адекватной формализации понятия электронного водяного знака (по крайней мере, авторам такое решение не известно), в данном разделе мы приводим единственный результат все из той же работы [6]. Следует особо подчеркнуть, что поскольку этот результат отрицательный, все приведенные выше критические замечания не применимы к определению 16. Чем шире класс алгоритмов Extract, о которых говорится в свойстве извлекаемости, тем отрицательный результат сильнее.

Теорема 7 ([6]). *Если существуют односторонние функции, то схемы электронных водяных знаков (в смысле определения* 16*) не существуют.*

Следствие 1 ([6]). *Эффективные схемы электронных водяных знаков (в смысле определения* 16*) не существуют.*

Список литературы

[1] *Варновский Н. П.* О теоретико-сложностном подходе к определению стойкости стеганографических систем. Сб. трудов 4-ой международной конференции «Дискретные модели в теории управляющих систем», 19–25 июля 2000 г. М.: «МАКС Пресс». 15–16.

[2] *Anderson R.* Stretching the limits of steganography. Proc. 1st Intern. Workshop on Inform. Hiding, 1996, LNCS, v. 1174, 39–48.

[3] *Anderson R.*, Petitcolas F. On the limits of steganography. IEEE J. on Selected Areas in Communications, 1998, v. 16, № 4, 474–481.

[4] *Cachin C.* An information-theoretic model for steganography. Proc. 2nd Intern. Workshop on Inform. Hiding, 1998, LNCS, v. 1525, 306–318.

[5] *Craver S.* On public-key steganography in the presence of an active warden. Proc. 2nd Intern. Workshop on Inform. Hiding, 1996, LNCS, v. 1525, 355–368.

[6] *Barak B.*, *Goldreich O.*, *Impagliazzo R.*, *Rudich S.*, *Sahai A.*, *Vadhan S.*, *Ke Yang*. On the (im)possibility of obfuscating programs. J. Kilian (ed.). Advances in Cryptology — Crypto'01. LNCS, v. 2139, Springer-Verlag, 2001, 1–18. См. электронную версию данной работы: `http://www.math.ias.edu/~boaz/Papers/obfuscate.ps`.

[7] *Diffie W.*, *Hellman M.* New directions in cryptography. IEEE Transactions om Information Theory, IT-22(6), 1976, 644–654.

[8] *Dedić N.*, *Itkis G.*, *Reyzin L.*, *Russell S.* Upper and lower bounds on black-box steganography. Cryptology ePrint Archive (`http//eprint.iacr.org`),

Report 2004/246.

[9] *Dodis Y.*, *Ong Sh. J.*, *Prabhakaran M.*, *Sahai A.* On the (im)possibility of cryptography with imperfect randomness. Proc. 45th Symp. Found. of Computer Science, 2004, 196–205.

[10] *Impagliazzo R.*, *Levin L.*, *Luby M.* Pseudo-random generation from one-way functions. Proc. 21st Symp. on Theory of Computing, 1989, 12–24.

[11] *Johnson N. F.*, *Jajodia S.* Steganalysis of images created using current steganography software. Proc. 2nd Intern. Workshop on Inform. Hiding, 1998, LNCS, v. 1525, 273–289.

[12] *Luby M.* Pseudorandomness and cryptographic applications. Princeton, NJ, Princeton University Press, 1996.

[13] *Pfitzmann B.* Information hiding terminology. Proc. 1st Intern. Workshop on Inform. Hiding, 1996, LNCS, v. 1174, 347–350.

[14] *Simmons G. J.* The prisoners' problem and the subliminal channel. Crypto'83, 1984, 51–67.

[15] *Zöllner J.*, *Federrath H.*, *Klimant H.*, *Pfitzmann A.*, *Piotraschke R.*, *Westfeld A.*, *Wicke G.*, *Wolf G.* Modeling the security of steganographic systems. Proc. 2nd Intern. Workshop on Inform. Hiding, 1998, LNCS, v. 1525, 344–354.

Математические проблемы обфускации

Н. П. Варновский, В. А. Захаров, Н. Н. Кузюрин

1. Введение

«Более практичный подход к проблеме поиска пары легко вычислимых взаимнообратных алгоритмов E и D таких, что зная E трудно найти D, использует трудность анализа программ на языках низкого уровня. Всякий, кто хоть раз пытался определить, каким образом работает программа на языке машинных команд, написанная кем-либо другим, знает, что сама программа E (т. е. что E делает) может быть очень трудна для понимания, исходя из алгоритма для E. Если программа преднамеренно запутывается путем добавления избыточных переменных и операторов, то поиск алгоритма для обратного преобразования может стать очень трудной задачей. Безусловно, программа E сама должна быть достаточно сложной, чтобы ее идентификация, исходя из пар вход-выход, была невозможна.

По-существу, требуется односторонний компилятор: он берет легко понимаемую программу, написанную на языке высокого уровня, и переводит ее в трудную для понимания программу на некотором машинном языке. Компилятор должен быть односторонним, поскольку требуется, чтобы компиляция выполнялась эффективно, но обращение этого процесса было трудной задачей.»

Данная цитата представляет собой не слишком хорошо известный фрагмент из основополагающей работы Диффи и Хеллмана [11]. Хотя само слово «обфускация» в этом фрагменте не встречается, в нем на неформальном уровне и, по-видимому, впервые изложена идея обфускирующего преобразования. Алгоритмы E и D, о которых здесь идет речь, — это алгоритмы шифрования и дешифрования соответственно криптосистемы с секретным ключом. Односторонний компилятор, о котором говорят Диффи и Хеллман, теперь принято называть обфускатором.

Следует заметить, что к моменту выхода в свет работы [11] никаких предложений по реализации идеи криптосистемы с открытым ключом не

Работа выполнена при поддержке гранта РФФИ № 03-01-00880.

было (криптосистемы RSA, Рабина и т. д. появились позже). Фактически, Диффи и Хеллман пытались обосновать возможность построения криптосистем с открытым ключом, предлагая взять какую-либо криптосистему с секретным ключом и подвергнуть обфускации ее алгоритм шифрования.

Неформально, обфускатором называется алгоритм $\mathcal{O}$, вообще говоря вероятностный, который получает на вход программу P и преобразует ее в программу $\mathcal{O}(P)$ таким образом, что выполняются следующие требования:

- (*функциональность*). Программы P и $\mathcal{O}(P)$ эквивалентны, т. е. вычисляют одну и ту же функцию.
- (*эффективность обфускации*). Накладные расходы на переход от P к $\mathcal{O}(P)$ (увеличение длины программы и времени ее выполнения) незначительны.
- (*стойкость*). Программа $\mathcal{O}(P)$ в каком-либо смысле трудна для понимания.

Вопрос о существовании обфускаторов, удовлетворяющих различным формализациям понятия стойкости, является основным предметом исследований по проблеме обфускации.

Программа $\mathcal{O}(P)$ называется обфускацией программы P. Иногда имеет смысл говорить об обфускации программы P безотносительно обфускатора. В таком случае все алгоритмические проблемы процесса обфускации игнорируются и просто ставится вопрос о существовании для данной программы P в ее классе эквивалентности такой программы, которая трудна для понимания.

Всюду далее делается различие между эффективностью обфускации, под которой понимается сформулированное выше свойство программы $\mathcal{O}(P)$, и эффективностью обфускатора $\mathcal{O}$.

Термин «обфускирующее преобразование» имеет два значения. Во-первых, так называют то преобразование, которое реализуется обфускатором. Во-вторых, обфускирующие преобразования — это набор достаточно простых эквивалентных преобразований программ, которые в совокупности образуют основу большинства известных на данный момент методов обфускации. В развернутой формулировке такие преобразования следует называть элементарными обфускирующими преобразованиями.

Сам термин «обфускация» латинского происхождения и заимствован из психиатрии, где в переводе на русский язык звучит как спутанность сознания или сумеречное состояние.

Список потенциальных приложений обфускации обширен, поэтому ограничимся лишь несколькими примерами.

- Преобразование криптосистем с секретным ключом в криптосистемы с открытым ключом, и схем аутентификации сообщений (MAC, имитовставки, имитоприставки) в схемы электронной подписи. Это приложение обсуждалось в приведенной выше цитате из Диффи и Хеллмана.
- Защита программного обеспечения. Подчеркнем, что речь идет не о защите от несанкционированного копирования, а о защите программ как интеллектуальной собственности в полном смысле этого слова. С нашей точки зрения основной интеллектуальной собственностью разработчика является не текст программы, а совокупность тех идей, которые положены в основу реализуемого ею алгоритма.
- Защита от атак на основе побочной информации. К таковой относится вся дополнительная информация, которую противник может собрать о процессе выполнения криптоалгоритма (время выполнения, потребляемая энергия, электромагнитное излучение и т. п.). Обфускация защищает от атак на основе побочной информации, поскольку не только дополнительная информация указанного типа, но и сама программа, реализующая криптоалгоритм, не должны давать противнику никакой полезной информации.
- Устранение случайных оракулов из криптографических протоколов. Модель со случайным оракулом позволяет доказывать стойкость многих криптографических протоколов, например, используемых на практике схем электронной подписи, для которых такие доказательства без предположения о случайном оракуле не известны. Появление обфускаторов с весьма высокой стойкостью позволило бы создавать программы, обладающие всеми необходимыми свойствами случайных оракулов.

Для читателя, хорошо знакомого с криптографией, уже одного этого списка может показаться достаточным, чтобы прийти к заключению, что обфускация невозможна. Но математика тем и отличается от всех иных сфер человеческой деятельности, в т. ч. и от других научных дисциплин, что и невозможность необходимо доказывать. Список же задач, которые могут быть решены с помощью обфускации, интересен еще и тем, что показывает, насколько сложна задача обфускации: она не проще самой сложной из задач этого списка.

Следует заметить, что существует еще и более практический подход к проблеме обфускации. Во многих работах задача обфускации понимается как противодействие алгоритмам декомпиляции программ, а обфускирующие преобразования сводятся к переименованию переменных и добавлению «мертвого кода». Это направление безусловно имеет право на существование, и может давать методы обфускации, пригодные

для некоторых приложений. Однако в тех же работах зачастую указываются потенциальные приложения обфускации наподобие приведенного выше списка. А это некорректно, поскольку для такого рода приложений требуются обфускаторы со стойкостью, не ниже чем обсуждаемая в настоящей статье.

И, наконец, последнее замечание. Как бы заманчиво не выглядели потенциальные приложения результатов того или иного научного исследования, наибольший интерес вызывают те задачи, которые возникли из потребностей развития самой науки. На наш взгляд, мотивацию для исследования проблемы обфускации доставляет, прежде всего, следующий общефилософский вопрос: может ли человек создать такую программу (осмысленную программу, делающую что-то полезное, а не случайный набор инструкций), которая была бы практически совершенно непонятна для любого другого.

2. Некоторые соглашения и обозначения

Обфускация представляет собой средство защиты программ. Всюду, где речь идет о защите информации, возникают два действующих лица. Один защищает информацию, а другой пытается эту защиту взломать. Они противоборствуют и являются друг для друга противниками. Но в литературе по защите информации термин «противник» всегда используется по отношению к взломщику.

Для вероятностной машины Тьюринга A и всякого входного слова x, $A(x)$ — случайная величина. Запись «для любого $A(x)$» означает квантор всеобщности по носителю случайной величины $A(x)$.

Для пары (детерминированных) машин Тьюринга A и B, $A \approx B$ обозначает их эквивалентность, т. е. для любого входного слова x, $A(x) = B(x)$. Это обозначение очевидным образом обобщается на случай, когда функции, вычисляемые машинами A и B, частичные: на всяком входе x либо обе машины не останавливаются, либо обе останавливаются и выдают одинаковое выходное слово.

Функция $\nu\colon \mathbb{N} \to [0, 1]$ называется пренебрежимо малой, если для любого $k \in \mathbb{N}$ существует n_0 такое, что для всех $n \geqslant n_0$, $\nu(n) < 1/n^k$. Для функции $\alpha\colon \{0, 1\}^* \to [0, 1]$ формула $\forall\, x \in \{0, 1\}^n\ \alpha(x) = \nu(n)$ используется как сокращение для следующего утверждения: существует функция $\beta\colon \mathbb{N} \to [0, 1]$ такая, что для всякого $x \in \{0, 1\}^n$, $\alpha(x) \leqslant \beta(n)$ и функция β пренебрежимо мала.

Для распределения вероятностей D на множестве X запись $x \in_D X$ означает выборку случайного элемента из множества X в соответствии с распределением вероятностей D. Обозначение $x \in_R X$ относится к частному случаю, когда распределение D равномерное. Через $\mathrm{Supp}\, D$ обозначается носитель распределения вероятностей D.

3. Теоретические предпосылки

Прежде чем приступать к исследованию вновь возникшей задачи, необходимо проанализировать все уже известные результаты, которые могут пролить свет на ее статус: является ли эта задача осмысленной, какова перспектива ее решения и т. д. В данном разделе обсуждаются некоторые математические результаты, которые, на наш взгляд, дают начальное представление о задаче обфускации.

3.1. Теория рекурсии

Всюду в данном подразделе, если не оговорено противное, вычислительная сложность задачи понимается в классическом теоретико-рекурсивном смысле. Например, какое-либо свойство программ считается не распознаваемым, если соответствующий предикат нерекурсивен.

Ответ на вопрос, существуют ли класс программ $\mathcal{P}$ и предикат π такие, что свойство π не распознаваемо для класса $\mathcal{P}$, хорошо известен. Примеры такого рода доставляют алгоритмически неразрешимые задачи, такие как проблема останова.

Более интересной, в связи с проблематикой обфускации, представляется классическая теорема Райса [15]. Для ее формулировки нам потребуются следующие обозначения.

Для машины Тьюринга M через $e(M)$ обозначается ее гёделев номер. Множество $\{0, 1\}^*$ рассматривается как множество двоичных представлений гёделевых номеров машин Тьюринга. Пусть $L \subseteq \{0, 1\}^*$ — подмножество такое, что если $M \approx M'$, (функции, вычисляемые машинами M и M', вообще говоря, частичные), то $e(M) \in L \iff e(M') \in L$, т. е. множество L замкнуто относительно эквивалентности машин Тьюринга.

Теорема 1 ([15]). *Множество L рекурсивно тогда и только тогда, когда оно тривиально, т. е. либо $L = \varnothing$, либо $L = \{0, 1\}^*$.*

Доказательство теоремы Райса можно найти, например, в статье Эндертона в «Справочной книге по математической логике» под редакцией Барвайса [5], а также в монографиях А. И. Мальцева [3] — гл. 4, п. 7.2, теорема 3, и Роджерса [4] — гл. 2, п. 2.1, упражнение 2-39(а).

На языке обфускации эту теорему можно переформулировать следующим образом. Множество L задает некоторое свойство программ (машин Тьюринга), инвариантное относительно эквивалентности. Если это свойство нетривиально, то для любого алгоритма A найдется

бесконечное множество $\mathcal{M}$ машин Тьюринга, для каждой из которых существует обфускация, т. е. для всякой $M \in \mathcal{M}$ найдется M', $M' \approx M$, такая, что алгоритм A на вопрос о принадлежности машины M' множеству L даст неверный ответ.

В связи с теоремой Райса возникает следующий вопрос: можно ли эту теорему «опустить» в какой-нибудь класс сложностей, соответствующий эффективным вычислениям, например, BPP. На уровне эффективных вычислений свойство программ считается трудно распознаваемым, если для соответствующего предиката нет полиномиального алгоритма. Аналог теоремы Райса, с заменой нерекурсивности на невычислимость за полиномиальное время, стал бы достаточно серьезным обоснованием возможности обфускации. Попытка получить такой аналог предпринималась [8], но к успеху не привела. Мы не обсуждаем возникшие при этом проблемы и отсылаем заинтересованного читателя к работе [8].

3.2. Теория выведывания

Вернемся еще раз к приведенной во введении цитате из Диффи и Хеллмана. В ней есть такие слова: «Безусловно, программа E сама должна быть достаточно сложной, чтобы ее идентификация, исходя из пар вход-выход, была невозможна». Проблемами, связанными с возможностью или невозможностью идентификации различных понятий, в т. ч. программ, исходя из пар вход-выход, занимается теория выведывания (learning theory) [16].

В самом общем виде задачи, рассматриваемые в теории выведывания, выглядят следующим образом. Пусть имеется некоторый класс понятий C. Примером может служить класс всех булевых формул в данном фиксированном базисе. Имеются два участника — выведыватель и противник. Последний выбирает некоторое понятие $c \in C$ и хранит его в секрете от выведывателя (но предполагается, что выведывателю известен тот класс C, из которого выбрано это понятие c). Далее выведыватель может получать от противника некоторую информацию о понятии c и его задача — идентифицировать в конце концов это понятие. Если существует алгоритм, позволяющий выведывателю сделать это для каждого $c \in C$, то класс C называется выведываемым, в противном случае — невыведываемым.

Замечание 4. При переходе на терминологию теории выведывания роли участников меняются. Тот участник «игры», который защищает программу с помощью обфускации, теперь называется противником, а его оппонент — выведывателем.

В приведенном выше неформальном описании не уточнялось, что понимается под словами «некоторая информация», «алгоритм», «идентификация понятия». Различные формализации этих терминов приводят к различным моделям выведывания.

Уже упоминавшаяся выше работа Вальянта [16] инициировала исследования, в которых акцент делается на поиске алгоритмов выведывания, работающих за полиномиальное (от параметров, определяемых моделью) время.

Замечание 5. Термин «теория выведывания» мы используем как рабочий за отсутствием лучшего варианта. Оригинальное название learning theory (дословно — теория обучения) нам представляется крайне неудачным. Во-первых, сам термин «обучение» является настолько общим и широким, что он не может быть адекватным названием отдельной, пусть даже весьма представительной, теории. Во-вторых, в англоязычной литературе выведывателя называют учеником (learner), что никак не согласуется с тем, что он обычно имеет дело с противником (здесь используется термин adversary, заимствованный из криптографии). Иногда противника называют учителем (teacher), но это уже просто противоречит его роли: как правило, вместо того чтобы помогать «ученику», такой «учитель» стремится усложнить ему задачу. Наконец, алгоритмы выведывания, в отличие от самообучающихся алгоритмов, на самом деле ничему не учатся. Это означает, что в результате любой попытки (успешной или неуспешной) выведать понятие c, выведывающий алгоритм не изменяет свое состояние и при следующей попытке выведать какое-либо понятие, даже если это будет то же самое c, начнет все сначала и повторит, в принципе, те же действия.

Существование обфускаций для какого-либо класса программ $\mathcal{P}$ теснейшим образом связано с его выведываемостью как класса понятий. Если существует эффективный алгоритм, который точно идентифицирует каждую программу из $\mathcal{P}$ на основе пар вход-выход, то никакую информацию об этих программах скрыть нельзя и ставить вопрос об их обфускации бессмысленно. Такие классы программ существуют и примеры строятся достаточно просто. В частности, можно рассмотреть класс программ, вычисляющих по какому-нибудь всем известному алгоритму значения полиномов фиксированной степени, скажем, над кольцом целых чисел. Для всякой конкретной программы $P \in \mathcal{P}$ выведывателю (играющему, в контексте задачи обфускации, роль противника) не известны только коэффициенты вычисляемого ею полинома. Ясно, что скрыть эту информацию невозможно.

Для того, чтобы задача обфускации была осмысленной, необходимо существование классов программ, которые являются в некотором,

достаточно сильном смысле, невыведываемыми. И такие классы программ существуют, причем простейшие примеры строятся достаточно легко. В частности, в литературе по обфускации популярен следующий пример. Пусть $\mathcal{P} = \{\mathcal{P}_n\}$, где $\mathcal{P}_n$ — множество всех программ $C_{\alpha,\beta}$. Здесь α, $\beta \in \{0, 1\}^n$ и на входе $x \in \{0, 1\}^n$ программа $C_{\alpha,\beta}$ выдает β, если $x = \alpha$, и 0^n — в противном случае. Функция, вычисляемая программой $C_{\alpha,\beta}$, не равна нулю в единственной точке α (разумеется, если $\beta \neq 0^n$). Такие функции называются дельта-функциями (англоязычный термин — point function). Алгоритм, реализуемый программой $C_{\alpha,\beta}$, считается одинаковым для всех α, β и общеизвестным, так что от выведывателя скрываются только параметры α и β. Выведыватель может выдавать произвольные запросы $x \in \{0, 1\}^n$ и получать ответы $C_{\alpha,\beta}(x)$. Ясно, что вероятность получить в ответ на отдельный запрос что-либо, отличное от 0^n, не выше 2^{-n}.

Классы программ, вычисляющих дельта-функции, с точки зрения перспектив обфускации представляют лишь ограниченный интерес. Во-первых, в большинстве случаев различия между функцией, не равной нулю лишь в отдельной точке, и тождественным нулем несущественны (хотя для некоторых специальных приложений, таких как парольная защита, подобные различия имеют принципиальное значение). Во-вторых, класс программ, вычисляющих дельта-функции, предельно узок, фактически он состоит из одной-единственной программы, в которой меняются только параметры. Задачу обфускации имеет смысл ставить только в том случае, если известны достаточно мощные классы программ, которые являются невыведываемыми как классы понятий. Примеры таких классов известны; отметим при этом, что их невыведываемость доказана, исходя из теоретико-сложностных или криптографических предположений. Ниже мы приводим примеры невыведываемых классов понятий. Для точной формулировки соответствующих результатов необходимо предварительно определить одну из моделей теории выведывания.

Пусть X — некоторое множество, которое будет рассматриваться как область определения понятий.

Классом представлений над множеством X называется пара (σ, C), где $C \subseteq \{0, 1\}^*$, а σ — отображение $\sigma\colon C \to 2^X$. Для всякой строки $c \in C$ множество $\sigma(c)$ называется понятием над X. Множество $\{\sigma(c) | c \in C\}$ называется классом понятий, а пара (σ, C) — его представлением.

Для класса представлений БУЛЕВЫ ФОРМУЛЫ, c — это запись некоторой булевой формулы в виде двоичной строки (предполагается, что выбран и зафиксирован некоторый канонический способ такой записи), C — множество таких записей для всех булевых формул,

$X = \{0, 1\}^*$. Отображение σ сопоставляет формуле, представленной строкой c и зависящей от n переменных, подмножество тех наборов из $\{0, 1\}^n$, на которых эта формула принимает значение 1. Допуская некоторую вольность речи, можно сказать, что $\sigma(c)$ — это булева функция, соответствующая формуле c, а класс понятий $\sigma(C)$ есть не что иное, как множество всех булевых функций. Ясно, что представление класса $\sigma(C)$ не единственно. Вместо множества всех булевых формул можно рассматривать, например, класс всех д. н. ф., класс всех к. н. ф. и т. д.

Нетрудно понять, что (эффективная) выведываемость или невыведываемость класса понятий зависит от выбранного представления. Например, если рассматривать представление булевых функций таблицами истинности, то любую функцию можно достаточно хорошо приблизить с помощью алгоритмов переборного типа. В то же время, в случае представления булевых функций произвольными булевыми формулами, этого сделать, при некоторых, достаточно правдоподобных предположениях, невозможно. Поэтому, на самом деле, выведываются не классы понятий, а классы представлений. Например, следовало бы говорить о выведываемости класса представлений БУЛЕВЫ ФОРМУЛЫ для класса понятий БУЛЕВЫ ФУНКЦИИ. Но поскольку такая терминология слишком громоздка, обычно все упоминания о классе представлений опускают и используют термины вида: класс понятий БУЛЕВЫ ФОРМУЛЫ. Кроме того, для упрощения терминологии часто опускают все упоминания об отображении σ и говорят о классе представлений C и о представлениях c из этого класса.

Классы представлений считаются параметризованными: $C = \bigcup_{n \geqslant 1} C_n$ и $X = \bigcup_{n \geqslant 1} X_n$. При этом если $c \in C_n$, то $\sigma(c) \subseteq X_n$. Параметр n естественно трактовать как некоторую меру сложности понятий. Для всякого понятия из C_n областью определения служит множество X_n. Например, для класса понятий БУЛЕВЫ ФОРМУЛЫ C_n — это множество всех формул, зависящих от n переменных.

Для всякого $c \in C_n$ пары вида $(x, 1)$, $x \in X_n$, называются положительными примерами для c, а пары вида $(x, 0)$, $x \in X_n$ — отрицательными примерами. Используется еще и следующая терминология: всякий элемент x множества X_n называется непомеченным примером или просто примером, а положительные и отрицательные примеры — помеченными примерами. Всякий помеченный пример есть пара $(x, c(x))$, где под $c(x)$ понимается значение характеристической функции множества $\sigma(c)$ в точке x.

В данной модели под выведывателем понимается некоторый алгоритм A, вообще говоря, рандомизированный.

Пусть C — параметризованный класс представлений и $c \in C_n$. Алгоритм предсказания с запросами принадлежности (membership queries) — это рандомизированный алгоритм A, который получает на вход три числа s, n и $\varepsilon > 0$, где число s представляет собой ограничение сверху на величину $|c|$. Противник моделируется посредством оракула O, который выбирает $c \in C_n$, а также распределение D на множестве X_n, и хранит их в секрете от A. Мы предполагаем, что алгоритм A работает с одним оракулом, но может выдавать запросы трех типов:

- Запрос принадлежности. Алгоритм A выбирает строку $x \in X_n$ и передает ее оракулу O, который возвращает значение $c(x)$.
- Запрос случайного помеченного примера. Оракул O не получает никаких входных данных и выдает пару $(x, c(x))$, где $x \in_D X_n$.
- Запрос случайного непомеченного примера. Оракул O не получает никаких входных данных и выдает строку y, где $y \in_D X_n$.

Алгоритм A может выдать любое количество запросов первых двух типов, после чего он должен выдать единственный запрос случайного непомеченного примера. Получив от оракула непомеченный пример y, алгоритм A уже не может делать никаких запросов к оракулу; он должен выдать свое предсказание $b \in \{0, 1\}$ и остановиться.

Определение 1. Алгоритм A^O является алгоритмом предсказания для класса C, если для любых чисел n, s и $\varepsilon > 0$, для любого $c \in C_n$ и для любого распределения D на множестве X_n

$$\Pr\{b \neq c(y)\} < \varepsilon.$$

Вероятность определяется распределением D и случайными величинами алгоритма A.

Класс представлений C называется полиномиально предсказываемым с запросами принадлежности, если для этого класса существует алгоритм предсказания, время работы которого ограничено некоторым полиномом от s, n и $1/\varepsilon$.

Следующая теорема доказана в работе [7].

Теорема 2. *В предположении вычислительной трудности любой из следующих трех задач*:

- *инвертирования функции шифрования криптосистемы RSA*;
- *распознавания квадратичных вычетов по составному модулю, который является произведением двух простых чисел одинаковой длины*;
- *факторизации чисел Блюма*,

класс понятий БУЛЕВЫ ФУНКЦИИ не является полиномиально предсказываемым с запросами принадлежности.

Мы сформулировали теорему 2 для класса БУЛЕВЫ ФУНКЦИИ, поскольку рассматриваем этот класс в качестве примера. На самом деле эта теорема верна и для некоторых других классов понятий [7]. В их числе — классы ПОРОГОВЫЕ СХЕМЫ КОНСТАНТНОЙ ГЛУБИНЫ и НЕДЕТЕРМИНИРОВАННЫЕ КОНЕЧНЫЕ АВТОМАТЫ.

Более подробную информацию о выведываемости и невыведываемости различных классов понятий можно найти в работе Англуин [6].

4. Основные определения и результаты

Из неформального описания обфускации, приведенного во введении, ясно, что обфускацию можно рассматривать как весьма специальный частный случай шифрования. Но имеются два существенных отличия. Во-первых, не требуется существования эффективного способа дешифрования «криптограммы». В этом смысле обфускация ближе к другому криптографическому примитиву — привязке к строке, который представляет собой обобщение привязки к биту (bit commitment). Во-вторых, «криптограмма» должна быть исполняемым программным кодом, функционально эквивалентным исходной программе. И это второе отличие оказывается весьма принципиальным. Как мы видели в подразделе 3.2, существуют классы программ, которые являются выведываемыми как классы понятий. Из всего сказанного можно сделать следующий важнейший вывод: в отличие от шифрования, которое универсально, т. е. применимо к любой информации, представленной, скажем, битовой строкой, обфускация заведомо не может защитить каждую программу.

Определение стойкости обфускации должно учитывать такое положение дел. На первый взгляд, определение стойкости можно сформулировать в следующем стиле: «Пусть $\mathcal{P}$ — класс программ, который является невыведываемым как класс понятий. Тогда обфускатор $\mathcal{O}$ для класса $\mathcal{P}$ называется стойким, если…». Но на данном пути возникают, по крайней мере, две трудности принципиального характера. Во-первых, в теории выведывания существуют различные модели и неясно, какую из них следует признать адекватной для использования в определении стойкости обфускации. Во-вторых, получится «нерабочее» определение, поскольку разделение классов понятий на выведываемые и невыведываемые представляет собой фундаментальную математическую проблему; на данный момент известен статус лишь очень немногих классов.

Более того, возможна и такая ситуация: класс программ является невыведываемым как класс понятий, но имеет некоторые критические

свойства, которые эффективно распознаваемы при доступе к программе как к оракулу (см. работу [12], в которой исследуется соотношение между выведываемостью и сложностью задачи распознавания свойств).

Авторы основополагающей работы по обфускации [8] справились со всеми обсуждавшимися выше трудностями, предложив определение стойкости, основанное на парадигме «черного ящика». Обфускатор $\mathcal{O}$ называется стойким, если текст программы $\mathcal{O}(P)$ дает любому эффективному алгоритму не больше информации о программе P, чем ее можно извлечь, наблюдая поведение программы P лишь на уровне входных и выходных слов. Доступ к парам вход-выход моделируется посредством оракула («черного ящика»). Оракул знает программу P и на запрос x (входное слово) отвечает выходным словом $P(x)$.

В работе [8] рассматриваются две формализации понятия программы — машина Тьюринга и булева схема, т. е. схема из функциональных элементов (которую в дальнейшем, для краткости, будем называть просто схемой).

Определение 2 ([8]). Вероятностный алгоритм $\mathcal{O}$ называется обфускатором для машин Тьюринга, если:

- (*функциональность*). Для всякой машины Тьюринга M любой выход $\mathcal{O}(M)$ алгоритма $\mathcal{O}$ на входе M описывает машину Тьюринга, которая вычисляет ту же функцию, что и M, т. е. $\mathcal{O}(M) \approx M$.
- (*эффективность обфускации*). Длина описания и время работы любой машины Тьюринга $\mathcal{O}(M)$ не более чем полиномиальны от соответствующих параметров машины M. Т. е. существует полином p такой, что для всякой машины Тьюринга M и любой $\mathcal{O}(M)$, $|\mathcal{O}(M)| \leqslant p(|M|)$, и если M останавливается через t шагов на некотором входе x, то $\mathcal{O}(M)$ останавливается на x через не более чем $p(t)$ шагов.
- (*стойкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любой машины Тьюринга M

$$\left|\Pr\{A(\mathcal{O}(M)) = 1\} - \Pr\left\{S^M\left(1^{|M|}\right) = 1\right\}\right| = \nu(|M|).$$

Определение 3 ([8]). Вероятностный алгоритм $\mathcal{O}$ называется обфускатором для схем, если:

- (*функциональность*). Для всякой схемы C всякий выход $\mathcal{O}(C)$ алгоритма $\mathcal{O}$ на входе C описывает схему, которая вычисляет ту же функцию, что и C.
- (*эффективность обфускации*). Существует полином p такой, что для всякой схемы C и всякой $\mathcal{O}(C)$, $|\mathcal{O}(C)| \leqslant p(|C|)$.

- (*стойкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любой схемы C

$$\left|\Pr\{A(\mathcal{O}(C)) = 1\} - \Pr\left\{S^C\left(1^{|C|}\right) = 1\right\}\right| = \nu(|C|).$$

Обфускатор $\mathcal{O}$ называется эффективным, если он работает за полиномиальное время.

Легко видеть, что требование стойкости в определении 2 эквивалентно следующему:

- (*стойкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любой машины Тьюринга M и любого предиката π

$$\left|\Pr\{A(\mathcal{O}(M)) = \pi(M)\} - \Pr\left\{S^M\left(1^{|M|}\right) = \pi(M)\right\}\right| = \nu(|M|).$$

Таким образом, ни для одного свойства π программы M текст программы $\mathcal{O}(M)$ не дает больше информации, чем доступ к «черному ящику». Аналогичное замечание справедливо и для определения 3.

Основной результат работы Барака и др. [8] отрицательный: стойкие обфускаторы не существуют.

Теорема 3 ([8]). *Обфускаторы для машин Тьюринга* (*в смысле определения* 2) *не существуют.*

Общая идея доказательства этой теоремы достаточно проста. Рассматривается семейство дельта-функций $C_{\alpha,\beta}$ (см. подраздел 3.2). Машина Тьюринга, вычисляющая функцию $C_{\alpha,\beta}$, для простоты обозначается таким же образом: $C_{\alpha,\beta}$. Определяется еще одно семейство машин Тьюринга $D_{\alpha,\beta}$. На входе C, где C — запись машины Тьюринга, $D_{\alpha,\beta}$ выдает 1, если $C(\alpha) = \beta$, и 0 — в противном случае. Сравниваются два сценария. В первом из них противник получает тексты программ $\mathcal{O}(C_{\alpha,\beta})$ и $\mathcal{O}(D_{\alpha,\beta})$. Он может просто выполнить $\mathcal{O}(D_{\alpha,\beta})(\mathcal{O}(C_{\alpha,\beta}))$, получить 1 и тем самым узнать, что существует входное слово, на котором машина $\mathcal{O}(C_{\alpha,\beta})$ выдает ненулевое выходное слово. Во втором сценарии моделирующей машине S предоставлен доступ к программам $C_{\alpha,\beta}$ и $D_{\alpha,\beta}$ как к оракулам. Ясно, что в этом случае отличить $C_{\alpha,\beta}$ от тождественно нулевой функции можно лишь с пренебрежимо малой вероятностью. Остается лишь преодолеть некоторые технические трудности, связанные с преобразованием, путем композиции, пары машин $C_{\alpha,\beta}$ и $D_{\alpha,\beta}$ в одну машину Тьюринга.

Аналогичным образом строится контрпример и в случае схем. Но при этом возникают технические трудности уже иного порядка, которые

авторы работы [8] преодолели с помощью криптографических примитивов. Именно из-за привлечения криптографических примитивов утверждение следующей теоремы условно.

Теорема 4 ([8]). *Если существуют односторонние функции, то обфускаторы схем (в смысле определения 3) не существуют.*

Для эффективных обфускаторов доказана [8] следующая лемма.

Лемма 1. *Если существуют эффективные обфускаторы, то существуют односторонние функции.*

Еще раз вспомним цитату из Диффи и Хеллмана: «…требуется односторонний компилятор».

Следствие 1 ([8]). *Эффективные обфускаторы для схем (в смысле определения 3) не существуют.*

До сих пор отрицательные результаты из работы [8] мы формулировали как утверждения о несуществовании обфускаторов. На самом деле из доказательств следует несколько более сильное утверждение о несуществовании обфускаций. Для математически строгой формулировки этого результата необходимо следующее определение.

Определение 4 ([8]). Необфускируемым семейством функций называется семейство $\{H_k\}_{k\in\mathbb{N}}$ распределений вероятностей H_k на множествах конечных функций (отображающих, скажем, множество $\{0,1\}^{n(k)}$ в множество $\{0,1\}^{m(k)}$), удовлетворяющее следующим требованиям:

- существует полином Q такой, что всякая функция из носителя распределения H_k вычислима схемой размера не более чем $Q(k)$. Более того, существует полиномиальный вероятностный алгоритм, который генерирует схемы в соответствии с распределением H_k.
- Существует предикат $\pi\colon \bigcup_{k\in\mathbb{N}} \mathrm{Supp}(H_k) \to \{0,1\}$ такой, что

1) $\pi(f)$ трудновычислим в случае доступа к функции f как к черному ящику: для любой полиномиальной вероятностной машины Тьюринга S

$$\left|\Pr\left\{S^f\left(1^k\right)=\pi(f)\right\}-1/2\right|=\nu(k);$$

2) $\pi(f)$ легко вычислим при наличии любой схемы, вычисляющей функцию f: существует полиномиальная машина Тьюринга A такая, что для всякой $f\in\bigcup_{k\in\mathbb{N}}\mathrm{Supp}(H_k)$ и для всякой схемы C вычисляющей f, $A(C)=\pi(f)$.

Теорема 5 ([8]). *Если существуют односторонние функции, то существуют необфускируемые семейства функций.*

Таким образом, существуют семейство функций $\{f\}$ и свойство (предикат) π такие, что никакая программа, вычисляющая функцию f, не скрывает это свойство (т. е. значение $\pi(f)$).

Теорема 5 может быть усилена в следующем направлении. Вместо предиката π рассматривается функция $\pi\colon \bigcup_{k\in\mathbb{N}}\mathrm{Supp}(H_k)\to\{0,1\}^*$, вычислимая за полиномиальное время. Требование 1, накладываемое на предикат π определением 4, заменяется требованием псевдослучайности строки $\pi(f)$. То есть никакая полиномиальная вероятностная машина Тьюринга, имеющая оракульный доступ к функции f, не может отличить строку $\pi(f)$ от чисто случайной строки той же длины. Семейства функций, удовлетворяющие таким образом модифицированному определению 4, называются полностью (totally) необфускируемыми семействами функций.

Теорема 6 ([8]). *Если существуют односторонние функции, то существуют полностью необфускируемые семейства функций.*

5. Проблема альтернативного определения стойкости

Отрицательные результаты, обсуждавшиеся в предыдущем разделе, побуждают исследователей искать возможности приемлемого ослабления требований определений 2 и 3. Таких направлений поиска может быть достаточно много; в данном разделе рассматриваются лишь некоторые из них.

5.1. Ограничение класса программ

Определения 2 и 3 говорят о произвольной машине Тьюринга и о произвольной схеме соответственно. Вместо этого можно ставить задачу обфускации лишь программ из некоторого более узкого класса $\mathcal{P}$. Проблема в том, что этот класс $\mathcal{P}$ должен иметь в своем определении достаточно конструктивности, чтобы с ним (классом) можно было работать.

Первый, вполне очевидный подход состоит в ограничении вычислительной сложности программ, входящих в класс $\mathcal{P}$. Но здесь следует учитывать следующий результат.

Теорема 7 ([8]). *В предположении вычислительной трудности задачи факторизации чисел Блюма, существует необфускируемое семейство функций, принадлежащее классу* TC^0.

Класс TC^0 состоит из функций, вычислимых схемами из пороговых элементов, имеющими полиномиальную сложность и константную глубину. Приведем формальное определение этого класса.

Рассматривается базис, состоящий из булевых функций И, ИЛИ, и MAJORITY с неограниченным количеством входных переменных.

Функция MAJORITY на входе $x_1, \ldots, x_k$ принимает значение 1, если количество единичных значений среди всех x_i больше, чем $k/2$, и 0 — в противном случае.

Определение 5. TC^0 — класс всех функций, вычислимых семействами схем (в указанном базисе) полиномиального размера и константной глубины.

Классы сложностей, находящиеся ниже класса TC^0 в иерархии сложностных классов, малы по объему и содержат лишь достаточно простые функции. Так что сужение класса $\mathcal{P}$ за счет теоретико-сложностных ограничений не выглядит перспективным.

Другой возможный подход — ограничить класс $\mathcal{P}$ лишь программами, имеющими одинаковое предназначение. Например, $\mathcal{P}$ — класс программ, реализующих алгоритмы шифрования криптосистем с секретным ключом. Этот и другие криптографические классы исследовались в работе [8]. Легко видеть, что полностью необфускируемые семейства функций позволяют строить контрпримеры и для всех таких классов. В частности, достаточно взять алгоритм шифрования какой-либо криптосистемы, стойкой против атаки с выбором открытого текста, «встроить» в него вычисление функции f из полностью необфускируемого семейства функций и выдавать, помимо криптограммы, значение $k \oplus \pi(f)$, где k — секретный ключ криптосистемы. Для противника, имеющего доступ к криптосистеме как к «черному ящику», модифицированная криптосистема останется стойкой. А из текста любой программы, вычисляющей модифицированную функцию шифрования, можно извлечь строку $\pi(f)$, а, следовательно, и секретный ключ k.

В математической криптографии имеется один фундаментальный результат, который также можно трактовать как аргумент в пользу невозможности обфускации алгоритмов шифрования. Речь идет о следующей теореме Импальяццо и Рудиха.

Теорема 8 ([13]). *Существует оракул, относительно которого существуют односторонние перестановки, но нет стойких криптосистем с открытым ключом.*

Здесь следует иметь в виду следующую криптографическую метатеорему:

Стойкие криптосистемы с секретным ключом существуют тогда и только тогда, когда существуют односторонние функции.

Это — метатеорема, поскольку точные формулировки и доказательства такого рода утверждений различаются в зависимости от определения конструкции криптосистемы и от рассматриваемых атак и угроз.

Таким образом, в универсуме, существование которого утверждается в теореме Импальяццо и Рудиха, существуют стойкие криптосистемы с секретным ключом, но нет стойких криптосистем с открытым ключом.

На первый взгляд может показаться, что этот результат слабее утверждения о невозможности обфускации алгоритмов шифрования из работы [8]. На самом деле эти результаты несравнимы. Первый из них [13] справедлив лишь относительно оракула, в то время как второй [8] — абсолютный. С другой стороны, в работе [8] показано лишь, что существуют (искусственно созданные) криптосистемы с секретным ключом, алгоритмы шифрования которых не допускают обфускации. Если же представить себе, что результат Импальяццо и Рудиха справедлив и в нерелятивизированной постановке (в том смысле, что для доказательства существования стойких криптосистем с открытым ключом требуется более сильное предположение, чем существование односторонних перестановок), то он означал бы, что:

- либо обфускация невозможна ни для какой криптосистемы с секретным ключом, стойкость которой доказана в предположении существования односторонних перестановок;
- либо такая обфускация возможна, но она сама требует предположения, более сильного, чем предположение о существовании односторонних перестановок, например, о существовании функций с секретом (на котором одном уже можно строить стойкие криптосистемы с открытым ключом).

5.2. Парадигма «серого ящика»

Требования к стойкости обфускации можно ослабить, если предоставить моделирующей машине S какую-либо дополнительную информацию, сверх той, которую эта машина может получить от «черного ящика». Один из возможных вариантов, модель с так называемым «серым ящиком», исследовался в работе [17]. Дополнительная информация, которую дает «серый ящик», это — трасса вычисления на данном входном слове. Такая модификация выглядит обоснованной, поскольку обфускация $\mathcal{O}(P)$ любой программы P не может быть «черным ящиком» в полном смысле этого слова. В самом деле, противник, получивший программу $\mathcal{O}(P)$, может для любого данного входного слова x получить не только выходное слово $P(x)$, но и всю трассу вычисления программы P на этом слове x. Всюду в данном подразделе обфускацию, стойкость которой определяется парадигмой «серого ящика», мы будем называть слабой.

Определение слабой обфускации отличается от определения 2 в двух аспектах. Первый — это спецификация оракула, используемого моделирующей машиной S. Когда машина S выдает оракулу запрос x, она получает в ответ не только выходное слово y, но также и трассу вычислений машины M на входном слове x. Заметим, что в данной модели существенно, какую из программ, эквивалентных машине M, использует оракул. В приводимом ниже определении предполагается, что оракул должен использовать исходную программу M. Это означает, что определение не требует, чтобы обфускатор скрывал какие-либо свойства, выведываемые эффективно, исходя из трасс вычислений машины M. Заметим также, что обфускатор может считаться стойким только в том случае, если он не открывает никакие свойства, которые остаются сокрытыми в трассах исходной программы.

Второй аспект — определение эквивалентности программ. В работе [17] рассматриваются программы с «памятью» (конечные автоматы), т. е. выход данной программы зависит не только от текущего входного слова, но также и от предшествующих.

Чтобы отличить оракул, используемый в определении 2, от оракула, предоставляемого моделирующей машине S в этой новой модели, последний обозначается через $\mathrm{Tr}(M)$. На входном слове x этот оракул выдает пару $(y, \mathrm{tr}_M(x))$, где y — выходное слово машины M на входе x, а $\mathrm{tr}_M(x)$ обозначает трассу вычисления машины M на этом входе.

Строка tr_M определяется как конкатенация всех последовательных инструкций, выполняемых машиной M на входе x.

Определение 6. Вероятностный алгоритм $\mathcal{O}$ называется слабым обфускатором для машин Тьюринга, если:

- (*функциональность*). Для всякой машины Тьюринга M всякий выход $\mathcal{O}(M)$ алгоритма $\mathcal{O}$ на входе M описывает машину Тьюринга, которая эквивалентна машине M в следующем смысле. Для всякой последовательности входов $x_1, \ldots, x_u$ соответствующие последовательности выходов машин $\mathcal{O}(M)$ и M совпадают, т. е., $\mathcal{O}(M)(x_i) = M(x_i)$ для всякого $i = 1, \ldots, u$.
- (*эффективность обфускации*). Длина описания и время выполнения всякой машины Тьюринга $\mathcal{O}(M)$ не более чем полиномиальны от соответствующих параметров машины M.
- (*стойкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любой машины Тьюринга M

$$\left|\Pr\{A(\mathcal{O}(M)) = 1\} - \Pr\left\{S^{\mathrm{Tr}(M)}\left(1^{|M|}\right) = 1\right\}\right| = \nu(|M|).$$

Теорема 9 ([17]). *Если существуют односторонние функции, то слабая обфускация невозможна.*

5.3. Классификация моделей обфускации на основе приложений

В работе [8] обсуждаются различные потенциальные приложения обфускации, в т. ч. защита программного обеспечения и преобразование криптосистем с секретным ключом в криптосистемы с открытым ключом. Но возникает вопрос о том, насколько определение 2 соответствует требованиям этих приложений. Например, если программа предназначена для продажи на рынке программного обеспечения, то она должна иметь «Руководство пользователя» или какой-либо другой документ, описывающий ее функциональные характеристики. Вряд ли найдется клиент, готовый заплатить деньги за «черный ящик». Поэтому в контексте защиты программного обеспечения как правило нужно скрывать не функцию, вычисляемую программой, а лишь некоторые особенности используемого алгоритма.

Если же возникает задача обфускации программы шифрования криптосистемы с секретным ключом (с целью преобразования в криптосистему с открытым ключом), то и функцию, и используемый алгоритм можно считать общеизвестными. А скрывать от противника требуется только секретный ключ.

Эти простые соображения подсказывают, что не может быть никакого универсального определения стойкости обфускации, и что такое определение должно зависеть от предполагаемых приложений.

Для исследования проблем обфускации в работе [17] предложены следующие три модели.

Обфускация для защиты программного обеспечения (*сокрытие алгоритма*). В данной постановке предполагается, что функция, вычисляемая программой, известна противнику. Наиболее очевидный путь формализации такой постановки состоит в фиксации некоторой программы P_0, эквивалентной исходной программе P, и предоставлении противнику P_0 в качестве дополнительного входного слова.

Например, если P — программа, осуществляющая полное раскрытие криптосистемы RSA, т. е. находящая ее секретный ключ по заданному открытому ключу, то P_0 может быть несложной для понимания программой, осуществляющей ту же угрозу путем полного перебора секретных ключей.

Моделирующая машина S, существование которой гарантируется определениями 2 и 3, также имеет доступ к тексту программы P_0. Модифицированное определение 2 выглядит следующим образом.

Определение 7. Вероятностный алгоритм $\mathcal{O}$ называется обфускатором для машин Тьюринга, если:

- Выполняются первые два требования (функциональность и эффективность обфускации) определения 2.
- (*стойкость*). Для всякой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для всех пар машин Тьюринга $(M, \widetilde{M})$, $M \approx \widetilde{M}$,

$$\left|\Pr\{A(\mathcal{O}(M), \widetilde{M}) = 1\} - \Pr\{S^M(1^{|M|}, \widetilde{M}) = 1\}\right| = \nu(|M|).$$

Машины A и S работают за время, ограниченное полиномом от длин их первых входов, $\mathcal{O}(M)$ и $1^{|M|}$ соответственно.

Заметим, что в том случае, когда алгоритм $\widetilde{M}$ эффективный, моделирующей машине S не требуется доступ к оракулу M.

Полная обфускация (*сокрытие функции*). Это — в точности понятие Барака и др. [8]. Пример потенциального приложения данного типа обфускации приведен во введении, а именно, исключение случайных оракулов из криптографических схем. Заметим, однако, что в данном случае обфускация не является полной. Противник знает априори, что случайный оракул должен вычислять некоторую функцию, которая отображает, скажем, множество $\{0, 1\}^{2n}$ в $\{0, 1\}^n$ и которая «выглядит» как случайная функция.

Сокрытие константы. В данной постановке, все что требуется скрыть от противника — это значение некоторой константы c_0, используемой программой P. Предполагается, что эта константа выбрана случайным образом из множества C достаточно большой мощности. Можно заведомо предполагать, что исходная программа P известна противнику полностью, за исключением константы c_0.

Пусть P — программа, которая использует константу c. Для простоты изложения мы предполагаем, что эта константа выбрана случайным равновероятным образом из множества $\{0, 1\}^n$. Для всякого заданного значения константы $c_0 \in \{0, 1\}^n$ через $P(c_0)$ мы обозначаем соответствующий вариант программы P. Длина программы P предполагается ограниченной полиномом от n.

Определение 8. Вероятностный алгоритм $\mathcal{O}$ называется обфускатором для машин Тьюринга, если:

- Выполняются первые два требования (функциональность и эффективность обфускации) определения 2.
- (стойкость) Для любой полиномиальной вероятностной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для всякой машины Тьюринга M и для любого значения константы $c_0 \in \{0, 1\}^n$

$$\left|\Pr\{A[\mathcal{O}(M)(c_0), M(c)] = 1\} - \Pr\left\{S^{M(c_0)}\left[1^{|M(c_0)|}, M(c)\right] = 1\right\}\right| = \nu(|M(c_0)|),$$

где $c \in_R \{0, 1\}^n$.

Заметим, что можно определить и более слабую форму сокрытия константы, в которой противник A и моделирующая машина S вместо выдачи одного бита должны угадать значение константы c_0.

Для некоторых ограниченных классов программ, при определенных криптографических предположениях, нетривиальное сокрытие константы существует (по крайней мере в слабой форме). В самом деле, всякая криптосистема с открытым ключом дает пример такого рода, поскольку ее программу шифрования можно рассматривать как скрывающую константу, а именно, секретный ключ.

6. Поиски оснований для оптимизма

Все приведенные выше результаты исследований по проблеме обфускации являются отрицательными. Такое положение дел может вызвать у читателя определенный скепсис по отношению к этому методу защиты информации. Здесь следует учитывать, что содержание данной статьи не отражает состояния исследований по проблеме обфускации в целом. Более того, если обратиться к литературе по обфускации, то имеется всего несколько работ с отрицательными результатами и на порядки больше публикаций, в которых предлагаются и анализируются различные обфускирующие преобразования. Но мы рассматриваем только те результаты и методы, которые имеют строгое математическое обоснование. А при таком ограничении соотношение между «положительными» и «отрицательными» результатами прямо противоположное, что и нашло отражение в настоящем обзоре. Причины, по которым «отрицательные» результаты преобладают, могут быть различными. Но, вероятно, основная состоит в том, что строить контрпримеры проще, чем математически строго обосновывать стойкость методов защиты информации.

В данном разделе мы кратко обсуждаем некоторые результаты, которые представляют собой первые попытки исследователей обосновать существование обфускаций.

Прежде всего заметим, что примеры классов программ, для которых существуют стойкие обфускаторы, скажем, в смысле определения 2, строятся довольно просто. Для этого достаточно взять класс

программ, который является выведываемым (точно идентифицируемым) как класс понятий (см. подраздел 3.2). В работе [14] такие обфускации называются тривиальными. Всюду ниже мы рассматриваем только нетривиальные обфускации.

6.1. О существовании обфускаторов

Известны лишь два результата, которые можно трактовать как некоторые (достаточно слабые) аргументы в пользу существования обфускаторов. Первый из них появился независимо и в немного отличающихся формулировках в нескольких работах. Речь идет о возможности «встраивания» в программу, предварительно преобразованную к специальному виду, NP-полной задачи. В результате задача распознавания некоторого свойства преобразованной программы становится NP-трудной (см., например, [10]). И хотя скрывается только одно свойство программы, а гарантируемая стойкость весьма слабая (мера сложности «в худшем случае»), результат имеет отношение именно к проблеме существования обфускаторов, поскольку преобразование применимо к широкому классу программ. Кроме того, следует иметь в виду, что подобные результаты являются необходимым этапом в процессе исследования проблемы обфускации.

Второй результат доказан все в той же работе [8]. В ней задача обфускации рассмотрена в универсуме, релятивизированном к оракулу $F\colon \{0,1\}^* \to \{0,1\}^*$. Поэтому все алгоритмы, о которых идет речь в определении обфускации, включая программу и алгоритм противника, имеют доступ к оракулу, вычисляющему функцию F. Релятивизации являются популярной темой исследований в теории сложности вычислений и в некоторых смежных областях математики. Но эта тема выходит за рамки данной статьи и поэтому здесь не обсуждается более подробно (некоторую дополнительную информацию можно найти в книгах [1], [2]).

Теорема 10 ([8]). *Существует оракул, относительно которого существуют эффективные обфускаторы схем.*

6.2. О возможности обфускации

Все известные результаты о существовании обфускаций относятся к дельта-функциям (см. подраздел 3.2) и по-существу говорят о сокрытии константы в смысле определения 8.

Из всех этих работ наиболее интересной является самая ранняя [9]. В ней само слово «обфускация» не встречается, однако рассматривается задача построения криптографического примитива, который мог бы заменить случайный оракул в криптографических протоколах. В качестве такого примитива предлагается так называемая схема хэширования (hashing scheme). Требования, накладываемые определением такой схемы, достаточны для обфускации дельта-функций.

Несколько более подробно, схема хэширования — это пара эффективных алгоритмов H, V. На входе x алгоритм H выбирает случайную строку r и вычисляет $H(x, r)$. Алгоритм V получает на вход x и c и проверяет, что c есть хэш-значение для x, т. е. что существует строка r такая, что $H(x, r) = c$.

Для обфускации дельта-функции достаточно вместо значения x, на котором эта функция не равна нулю, хранить его хэш-значение $H(x, r)$. Для любого входа y алгоритм V позволяет проверить равенство $x = y$. Рандомизированное хэширование позволяет скрыть всю частичную информацию о значении x в нижеследующем смысле. Пусть I_x — оракул, который на входе z выдает 1, если $z = x$, и 0 — в противном случае. Основу определения схемы хэширования составляет следующее требование стойкости:

- для всякой полиномиальной машины Тьюринга A существует полиномиальная вероятностная машина Тьюринга S такая, что для любого семейства распределений $\{X_k\}$ (распределение вероятностей X_k заданно на множестве $\{0,1\}^k$) и любого предиката π, вычислимого за полиномиальное время,

$$\left|\Pr\{A(H(x, r)) = \pi(x)\} - \Pr\left\{S^{I_x}\left(1^{|x|}\right) = \pi(x)\right\}\right| = \nu(k),$$

где значение x выбрано случайным образом из распределения X_k. Ясно, что по существу это — требование стойкости обфускирующего преобразования.

Заметим, что в схемах хэширования коллизии возможны, хотя и с пренебрежимо малой вероятностью. Поэтому обфускирующие преобразования, основанные на схемах хэширования, сохраняют функциональность лишь приближенно.

В работе [9] схема хэширования построена, исходя из нестандартного и очень сильного предположения о вычислительной сложности распознавательного варианта задачи Диффи — Хеллмана.

Обфускация дельта-функций может быть основана и на стандартном криптографическом предположении о существовании односторонних перестановок [18]. Но при этом гарантированно (т. е. доказуемо) скрывается только одно свойство функции: является ли она дельта-функцией или тождественным нулем.

В статье [14] доказано существование обфускаций дельта-функций в модели со случайным оракулом. Этот результат, однако, не может не

наводить на некоторые размышления о замкнутом круге. Ведь одной из основных мотиваций для исследования проблемы обфускации является именно возможность устранения случайных оракулов из криптографических конструкций.

Хотя работа [14] называется «Положительные результаты и методы для обфускации», наиболее интересный ее результат отрицательный. Авторы исследовали вопрос о возможности композиции обфускаций в следующей постановке. Пусть даны два класса программ $\mathcal{F}$ и $\mathcal{G}$ и известно, что для обоих классов существуют обфускации. Что можно сказать о существовании обфускаций для класса $\mathcal{A}$, состоящего из композиций программ из $\mathcal{F}$ и $\mathcal{G}$? Композицию программ можно понимать в различном смысле. Если композиция определяется как суперпозиция функций, то класс $\mathcal{A}$ состоит из программ вида $A = F \cdot G$, $F \in \mathcal{F}$, $G \in \mathcal{G}$. Доказано, что если задача факторизации чисел Блюма является вычислительно трудной, то существуют классы обфускируемых программ $\mathcal{F}$ и $\mathcal{G}$, для которых класс $\mathcal{A}$ реализует необфускируемое семейство функций. Этот результат выводится из теоремы 7. Таким образом, при композиции программ свойство обфускируемости, вообще говоря, не сохраняется.

В заключение скажем несколько слов о работе [19], которая вышла в свет, когда данная статья уже готовилась к печати. В работе [19] также рассматриваются обфускации для дельта-функций. Требование стойкости обфускации сформулировано в следующем модифицированном виде.

- (*стойкость*). Для всякого семейства $\{A_n\}$ схем полиномиального размера и всякой функции $\varepsilon(n) = 1/n^{O(1)}$ существует семейство вероятностных схем $\{S_n\}$ полиномиального от $(n, 1/\varepsilon)$ размера такое, что для всех достаточно больших n и для всякой схемы $C \in \mathcal{C}_n$

$$\left|\Pr\{A_n(\mathcal{O}(C)) = 1\} - \Pr\left\{S_n^C\left(1^{|C|}\right) = 1\right\}\right| \leqslant \varepsilon(n).$$

Здесь под программой понимается булева схема, рассматривается класс $\mathcal{C}$ таких схем, а через $\mathcal{C}_n$ обозначается подмножество $\mathcal{C}$, состоящее из всех схем с n входами.

Эта формулировка отличается от требования стойкости в определении 3 в двух существенных аспектах. Во-первых, рассматривается неоднородная модель вычислений, т. е. и противник A, и моделирующий алгоритм S являются семействами булевых схем. Во-вторых, размер схемы S_n ограничен полиномом от $1/\varepsilon$, и, следовательно, в случае пренебрежимо малой функции ε, моделирующий алгоритм может иметь сверхполиномиальную сложность.

Исходя из нестандартного предположения о существовании перестановок, односторонних в очень сильном смысле, для семейства дельта-функций доказано существование обфускаций, стойких относительно модифицированного определения. Нестандартность предположения о перестановке заключается в следующем требовании: для всякого полинома p и для всякого семейства $\{A_n\}$ схем размера $p(n)$ существует полином q такой, что мощность множества входов длины n, на которых схема A_n инвертирует перестановку, не превосходит $q(n)$.

Список литературы

[1] *Гэри М.*, *Джонсон Д.* Вычислительные машины и труднорешаемые задачи. М.: Мир, 1982.

[2] *Китаев А. Ю.*, *Шень А. Х.*, *Вялый М. Н.* Классические и квантовые вычисления. М.: МЦНМО, 1999.

[3] *Мальцев А. И.* Алгоритмы и рекурсивные функции. 2-е изд. М.: Наука, 1986.

[4] *Роджерс Х.* Теория рекурсивных функций и эффективная вычислимость. М.: Мир, 1972.

[5] *Эндертон Г.* Элементы теории рекурсии. В кн. Справочная книга по математической логике. Дж. Барвайс (ред.). Т. III. Теория рекурсии. М.: Наука, 1982, 9–50.

[6] *Angluin D.* Computational learning theory: survey and selected bibliography. STOC'24, 1992, 351–369.

[7] *Angluin D.*, *Kharitonov M.* When won't membership queries help? STOC'23, 1991, 444–454.

[8] *Barak B.*, *Goldreich O.*, *Impagliazzo R.*, *Rudich S.*, *Sahai A.*, *Vadhan S.*, *Ke Yang*. On the (im)possibility of obfuscating programs. J. Kilian (ed.). Advances in Cryptology — Crypto'01. LNCS, v. 2139, Springer-Verlag, 2001, 1–18. См. электронную версию данной работы: `http://www.math.ias.edu/~boaz/Papers/obfuscate.ps`.

[9] *Canetti R.* Towards realizing random oracles: Hash functions that hide all partial information. Crypto'97, 455–469.

[10] *Chow S.*, *Gu Y.*, *Johnson H.*, *Zakharov V.* An approach to the obfuscation of control flow of sequential computer programs. Information Security conference, LNCS, v. 2200, 2001, 144–156.

[11] *Diffie W.*, *Hellman M.* New directions in cryptography. IEEE Transactions om Information Theory, IT-22(6), 1976, 644–654.

[12] *Goldreich O.*, *Goldwasser S.*, *Ron D.* Property testing and its connection to learning and approximation. FOCS'37, 1996, 339–348; см. также `http://theory.lcs.mit.edu/~oded/ggr.html`.

[13] *Impagliazzo R.*, *Rudich S.* Limits on the provable consequences of one-way permutations. STOC'21, 1989, 44–61.

[14] *Lynn B.*, *Prabhakaran M.*, *Sahai A.* Positive results and techniques for obfuscation. EUROCRYPT'2004, LNCS, v. 3027, 2004, 20–39.
[15] *Rice H.* Classes of recursively enumerable sets and their decision problems. Trans. Amer. Math. Soc., 74, 1953, 358–366.
[16] *Valiant L.* A theory of learnable. Comm. ACM, 1984, v. 27, № 11, 1134–1142.
[17] *Varnovsky N.* A note on the concept of obfuscation. Труды Института системного программирования: Том 6. Под ред. В. П. Иванникова. — М.: ИСП РАН, 2004, 127–136.
[18] *Varnovsky N.*, *Zakharov V.* On the possibility of provably secure obfuscating programs. Proc. 5th Conf. Perspectives of System Informatics, 2003, 71–78.
[19] *Wee H.* On obfuscating point functions. Cryptology ePrint Archive (`http://eprint.iacr.org`), Report 2005/001.

STOC = Proceedings of the Annual ACM Symposium on Theory of Computing. New York: ACM Press.

FOCS = Proceedings of the Annual Symposium on the Foundations of Computer Science. Los Alamitos, CA: IEEE Computer Society Press.

Математические модели распределенных компьютерных систем

В. А. Васенин, А. В. Галатенко

В настоящей работе рассматриваются математические модели гарантированно защищенных систем, строится формальное определение безопасности и приводятся просто проверяемые условия, гарантирующие безопасность. Также рассматриваются ограничения моделей и модели, ориентированные на скрытую компрометацию систем.

1. Введение

При анализе защищенности информации, обрабатываемой средствами вычислительной техники, принято выделять следующие типы внутренних и/или внешних угроз:

- угроза конфиденциальности информации;
- угроза целостности информации;
- угроза доступности информации.

Как правило, при построении математических моделей основное внимание уделяется первым двум типам, тогда как угрозы третьего типа в моделях обычно не рассматриваются. В значительной степени это связано с трудностью формализации понятия доступности.

Изначально проверка безопасности компьютерных систем была устроена следующим образом. В течение заданного промежутка времени группа экспертов пыталась скомпрометировать систему, и, если это удавалось, система признавалась уязвимой, в противном случае делался вывод о безопасности. Такой подход оказался недостаточно эффективным, так как не гарантировал, что злоумышленник не применит атаку, не использованную экспертами, которая может оказаться успешной. В 1970-х годах начал развиваться доказательный подход, заключающийся в построении математической модели компьютерной системы, формальном определении понятия безопасности, нахождении легко проверяемых достаточных условий, обеспечивающих безопасность, и проверке выполнимости найденных условий в модели-

руемой системе. Было создано большое количество моделей, среди которых можно выделить модель Белла — Лападула [1], модель Дороти Деннинг [2], модель Take-Grant [3], модель Biba [3], модель Low-watermark [3], а также класс моделей невлияния. В настоящее время, пожалуй, модели невлияния наиболее актуальны как в теоретическом, так и в практическом плане.

2. Модели невлияния

Рассмотрим простейший вариант модели невлияния, ориентированной на защиту секретности, для системы с двумя классами пользователей, заданный детерминированным конечным автоматом [4].

Под автоматом понимается тройка $V = (A, Q, \varphi)$, где A и Q — конечные множества (входной алфавит и множество состояний, соответственно), $\varphi\colon A^* \times Q \to Q$ — функция переходов (см. [5]). Перечислим допущения о компьютерной системе, моделируемой автоматом V. Пусть в системе работают два пользователя: H и L. Пользователь H имеет право знать о системе все, а L — только часть информации. Под безопасностью для V содержательно понимаем ситуацию, когда пользователь L не замечает влияния H на V. Формализуем данное определение. Пусть $A = A_L \sqcup A_H$, то есть у каждого пользователя есть свой набор команд, не пересекающийся с набором другого пользователя. Это предположение не нарушает общности, так как под командой можно понимать пару (идентификатор пользователя, команда). В каждый момент времени только один пользователь может вводить команды. Пусть состояние — вектор параметров, часть которых соответствует пользователю L, а оставшаяся часть — пользователю H. Исходя из описанной логики, формально определим понятие безопасности.

Введем на Q отношение эквивалентности. Пусть $q_1, q_2 \in Q$. Будем считать, что $q_1 \sim q_2$, если они совпадают на компонентах, соответствующих L. Естественным образом определяется проектор $\pi\colon Q \to Q/\sim$. Рассмотрим функцию $F\colon A^* \to {A_L}^*$, определенную следующим образом. Пусть $\alpha \in A^*$, $\alpha = a_1 a_2 \ldots a_k$, тогда $F(\alpha) = F(a_1)F(a_2) \ldots F(a_k)$. Пусть $a \in A$. Тогда $F(a) = \lambda$ (λ — пустой символ), если $a \in A_H$, и $F(a) = a$ в противном случае.

Скажем, что H не влияет на L, если выполнены следующие два условия:

1) отображение $\widehat{\varphi}\colon {A_L}^* \times Q/\sim \to Q/\sim$, заданное формулой $\widehat{\varphi}(\alpha_L, [q]) = [\varphi(\alpha_L, q)]$ (под $[q]$ понимается класс состояний, эквивалентных q) корректно определено;
2) следующая диаграмма коммутативна:

$$\begin{array}{ccc} A^* \times Q & \xrightarrow{\varphi} & Q \\ \downarrow{\scriptstyle F\times\pi} & & \downarrow{\scriptstyle \pi} \\ (A_L)^* \times Q/\sim & \xrightarrow{\widehat{\varphi}} & Q/\sim \end{array}$$

Будем говорить, что система безопасна, если в ней выполнены условия 1 и 2. Содержательно это означает, что информационные потоки в системе направлены снизу вверх (запрещено читать вверх и писать вниз), таким образом утечка секретных данных становится невозможной. Приведем простые условия, гарантирующие безопасность системы (такие условия называются «раскруткой» (unwinding)).

Теорема ([4]). *Система безопасна тогда и только тогда, когда условия* 1 *и* 2 *выполнены на словах длины* 1.

Замечание: если пользователей L и H поменять местами, получится модель, ориентированная на защиту целостности.

Рассмотрим некоторые обобщения описанной простейшей модели. Очевидным образом такое обобщение производится на случай произвольной конечной решетки меток пользователей (в этом случае условия невлияния могут не выполняться только для пользователей, метки которых сравнимы, и метка влияющего пользователя меньше или равна метке пользователя, на которого оказывается влияние). Заметим, что отношение невлияния является транзитивным, поэтому достаточно проверить выполнение условий теоремы только для соседних элементов решетки.

Довольно очевидно производится и обобщение на случай недетерминированных конечных автоматов [4]. Менее тривиальным обобщением является случай вероятностных автоматов [6]. Важность недетерминированного и вероятностного обобщения определяется высокой сложностью современных систем, вынуждающей снижать число состояний и таким образом огрублять модель. Во всех перечисленных моделях условия безопасности «раскручиваются» до условий на переходы за один такт, предоставляя эффективные критерии или достаточные условия проверки безопасности.

Выделим обобщение модели на сетевой случай, когда происходит объединение двух подсистем, в каждой из которых реализована модель невлияния с некоторой решеткой меток [7]. В этом случае оказывается, что при достаточно общих предположениях о сетевом взаимодействии (самым строгим является предположение о двустороннем характере сетевого обмена, но такое требование выполнено для подавляющего большинства современных сетевых протоколов) безопасное,

не нарушающее политики безопасности в подсистемах, взаимодействие возможно тогда и только тогда, когда решетки подключенных к сети субъектов изоморфны, и существует изоморфизм, при котором сетевой обмен возможен только между переходящими друг в друга классами. Данное обобщение позволяет проводить декомпозицию системы, строя общую непротиворечивую политику безопасности на основе политик подсистем.

3. Унификация политик безопасности

Выше был рассмотрен случай унификации двух многоуровневых политик безопасности. Рассмотрим еще два распространенных класса политик — основанных на дискреционном разграничении доступа и основанных на ролевой модели [8]. В работе [9] показывается, что справедливы следующие утверждения:

• Пусть S_1 — система с дискреционным разграничением доступа, S_2 — система с ролевым разграничением доступа. Тогда политика безопасности в S_1 может быть выражена в терминах политики в S_2.

• Пусть S_1 — система с ролевым разграничением доступа без административных ролей, S_2 — система с дискреционным разграничением доступа. Тогда политика безопасности в S_1 может быть выражена в терминах политики в S_2 без введения дополнительных видов доступа.

• Пусть S_1 — система с ролевым разграничением доступа, S_2 — система с дискреционным разграничением доступа. Тогда политика безопасности в S_1 может быть выражена в терминах политики в S_2, возможно, с введением дополнительных видов доступа.

Отметим, что приведенные утверждения доказываются конструктивно. Таким образом, имеется алгоритм, позволяющий сводить описанные классы политик друг к другу. Многоуровневая политика безопасности очевидным образом также может быть выражена с помощью политики с дискреционным разграничением доступа (следовательно — и с ролевым разграничением доступа). Однако общего результата, проверяющего возможность унификации двух политик и конструктивно задающего объединяющую политику, пока не получено.

4. Ограничения моделей

При построении упомянутых ранее математических моделей защищенных систем неявно использовалось два следующих предположения.

• Абсолютно надежная аутентификация (если, например, пользователь L может войти в систему под именем пользователя H, о невлиянии говорить просто бессмысленно).

• Абсолютно надежная реализация (если, например, из-за сбоя программного обеспечения вместо отказа в обслуживании доступ к ресурсам системы будет предоставлен, теоретические построения окажутся несостоятельными).

В области информационной безопасности имеют место две противоположные тенденции:

- снижение уровня защищенности, обеспечиваемого существующими реализациями, как в силу выявления уязвимостей и появления новых методов атак, так и вследствие повышения быстродействия компьютеров и расширения возможностей применения метода «грубой силы»;
- появление новых реализаций сервисов безопасности, обладающих повышенной защищенностью.

Уровень защищенности, обеспечиваемый некоторой реализацией сервиса безопасности,со временем имеет тенденцию к снижению. Выявляются дефекты реализации, порой обнаруживаются просчеты алгоритмического плана, за счет повышения быстродействия электронных компонентов становятся применимыми методы грубой силы. Как следствие, при создании надежной системы аутентификации важную роль играют архитектурные аспекты. Возникает необходимость реализации централизованной, целостной, динамически модифицируемой, модульной системы управления аутентификацией. В качестве решения может рассматриваться система встраиваемых неинтерактивных модулей аутентификации [10], [11]. В компьютерных программах (в том числе — в компонентах системы обеспечения безопасности) периодически обнаруживаются ошибки. Некоторые из них могут приводить к компрометации системы. Можно выделить следующие классы ошибок, критических с точки зрения безопасности: 1) переполнения буферов; 2) переполнения целых чисел; 3) нецелостность привилегированных транзакций; 4) некорректная обработка привилегий; 5) некорректная обработка форматированного ввода/вывода.

Наиболее часто эксплуатируются ошибки первых трех типов. Так, в ядре ОС Linux была обнаружена ошибка типа 3, позволяющая локальным пользователям получать привилегии суперпользователя [12]. В популярной реализации протокола удаленного доступа SSH обнаруживались ошибки типа 1 и 2, позволяющие неавторизованному пользователю входить в систему [13]. Легко увидеть, что комбинация таких ошибок позволяет любому злоумышленнику получить неограниченный доступ к системе.

Существует ряд подходов, в той или иной степени препятствующих появлению и/или эксплуатации ошибок в программном обеспечении.

Оптимальным вариантом представляется формальное доказательство правильности работы программы [14]. Однако провести такое доказательство удается далеко не для всех программ. Упрощением задачи является формальное доказательство отсутствия в программе ошибок заданного класса (например, в [15] рассматривается задача выявления переполнений буферов). В обоих случаях для анализа необходим исходный текст программ.

Имеется еще несколько подходов к анализу исходных текстов программ на предмет выявления ошибок. Значительное число программистов придерживается точки зрения, в соответствии с которой программа не содержит ошибок в том и только том случае, когда ее правильность «очевидна». При этом понятие очевидности не является строго определенным. Некоторые средства анализа просматривают исходный текст программ и выявляют «опасные» места, например функции, в которых возможна некорректная обработка входных аргументов, или неатомарные операции. В некоторых случаях применение описанных подходов позволяет улучшить качество программ, однако в общем случае получаемые результаты не являются ни необходимыми, ни достаточными для безошибочности программ.

Широко используемым средством обнаружения ошибок (и оценки числа ошибок) является тестирование [16]. Оптимальным вариантом было бы полное тестирование, однако в силу ряда причин (например, большого объема тестов, изменений и доводки программы в процессе тестирования) приходится проводить лишь частичное тестирование, позволяющее в рамках некоторой математической модели оценивать число невыявленных ошибок. Отметим, что важно проводить не просто тестирование на корректных данных, но и тестирование на некорректных данных, и стресс-тестирование [16]. Существенным преимуществом тестирования по сравнению с доказательством правильности является отсутствие необходимости иметь исходные тексты программ.

Выявлять ошибки можно и динамически, в процессе выполнения программ. Имеется широкий класс систем, ориентированных на обеспечение корректности работы с памятью и стеком в процессе выполнения программы (выделим свободно распространяемую систему valgrind (citevaseningalatenko:val, позволяющую обнаруживать широкий класс ошибок, прежде всего — связанных с памятью). Существенным недостатком средств данного класса является значительное замедление работы программы. Средства активного аудита [18] также могут рассматриваться как средства динамического выявления ошибок, так как эксплуатация ошибок (уязвимостей) часто приводит к аномальному поведению системы в целом или отдельных ее компонент.

5. Скрытая компрометация

Пусть в компьютерную систему заложена вредоносная подсистема (например, на аппаратном уровне, на этапе создания). Обозначим основную систему через L, вредоносную подсистему — через H. Если выполнены условия невлияния H на L, вредоносная подсистема окажется невидимой для пользователей основной системы.

Если используется оборудование общего назначения, то разумно предположить, что вредоносная подсистема нуждается в дополнительных указаниях (например, где именно искать необходимые данные, как отделить интересующие данные и др.). Возникает необходимость коммуникаций между удаленным управляющим устройством и вредоносной подсистемой. Для решения такой задачи можно воспользоваться скрытыми каналами передачи информации [19]. Как показано в [20] и [21], при достаточно общих предположениях (например, при наличии синонимичных конструкций в протоколе обмена информации) существует скрытый канал, не обнаруживаемый наблюдателем.

6. Заключение

В заключении перечислим ряд задач, решение которых представляется весьма актуальным:

- построение математических моделей систем, ориентированных на поддержание высокой доступности;
- рассмотрение более сложных, вообще говоря неатомарных, доступов;
- разработка методов редукции и декомпозиции моделей больших распределенных систем;
- создание средств формализации компьютерных атак, позволяющих автоматически проверять устойчивость распределенных компьютерных систем к этим атакам.

Список литературы

[1] *McLean J.* Reasoning about security models. Proc. of the 1987 IEEE Computer Society Symposium on Research in Security and Privacy, pp. 123–131, Oakland, CA, 1987.

[2] *Denning D.* A Lattice Model of Secure Information Flow. Comm. of the ACM, Vol.19, № 5, May 1976, `http://www.cs.georgetown.edu/~denning/infosec/lattice76.pdf`.

[3] *Грушо А. А., Тимонина Е. Е.* Теоретические основы защиты информации. «Яхтсмен», 1996.

[4] *Moskowitz I. S.*, *Costich O. L.* A Classical Automata Approach to Noninterference Type Problems. Department of the Navy, Naval Research Laboratory, 1992, http://chacs.nrl.navy.mil/publications/CHACS/1992/1992moskowitz-foundations.pdf.
[5] *Кудрявцев В. Б.*, *Алешин С. В.*, *Подколзин А. С.* Введение в теорию автоматов. М.: Наука, 1985.
[6] *Галатенко А. В.* Об автоматной модели защищенных компьютерных систем. Интеллектуальные системы, т. 4, № 3–4. М., 1999. С. 263–270.
[7] *Галатенко А. В.* Об автоматном управлении регулирования доступа к распределенным данным. Материалы конференции «Проблемы теоретической кибернетики», Москва, 2003.
[8] *Sandhu R. S.*, *Coyne E. J.*, *Feinstein H. L.*, *Youman C. E.* Role-Based Access Control Models. Computer, pp. 38–47, February 1996.
[9] *Андреев О. О.* Сравнение ролевой и дискреционной моделей разграничения доступа. Представлено в настоящем сборнике.
[10] *Галатенко А. В.*, *Наумов А. А.*, *Слепухин А. Ф.* Реализация системы управления доступом к информации в виде встраиваемых модулей аутентификации. Материалы конференции «МаБИТ-2003», Москва, 2003, с. 237–239.
[11] *Галатенко А. В.* Реализация сервисов безопасности на основе встраиваемых модулей. Информационная безопасность. Инструментальные средства программирования. Микропроцессорные архитектуры. Москва, НИИСИ РАН, 2003, с. 91–106.
[12] Материалы по ошибке в ядре ОС Linux. http://www.securityfocus.com/archive/1/315635.
[13] Материалы по ошибкам в реализации сервера протокола SSH. http://openssh.org/security.html.
[14] *Кларк Э. М.*, *Грамберг О.*, *Пелед Д.* Верификация моделей программ: model checking. М.: МЦНМО, 2002.
[15] *Галатенко А. В.*, *Пучков Ф. М.*, *Шапченко К. А.* Способ анализа программ на наличие угроз переполнения буферов. Материалы конференции «Информационная безопасность регионов России», С-Пб., 2003, С. 33.
[16] *Липаев В. В.* Программно-технологическая безопасность информационных систем. JetInfo, № 6–7, 1997, http://www.jetinfo.ru/1997/6-7/1/article1.6-7.1997.html.
[17] Материалы по системе valgrind. http://valgrind.kde.org.
[18] *Галатенко А. В.* Активный аудит. JetInfo, № 8, 1999, http://www.jetinfo.ru/1999/8/1/article1.8.1999.html.
[19] *Тимонина Е. Е.* Скрытые каналы (обзор). JetInfo, № 11, 2002, http://www.jetinfo.ru/2002/11/2002.11.pdf.
[20] *Грушо А. А.* Скрытые каналы и безопасность информации в компьютерных системах. Дискретная математика, т. 10, вып. 1, 1998, с. 3–9.
[21] *Грушо А. А.* О существовании скрытых каналов. Дискретная математика, т. 11, вып. 1, 1999, с. 24–28.

Математическое и программное обеспечение активного аудита больших распределенных систем

В. А. Васенин, А. В. Галатенко, В. В. Корнеев, А. А. Макаров

Введение

Массовое внедрение микропроцессоров, их перманентно возрастающие возможности и быстрый рост числа практически значимых систем на базе микропроцессоров привели к тому, что функции управления и обеспечения безопасности информационно-телекоммуникационных (далее по тексту инфо-телекоммуникационных) систем все чаще стали автоматизироваться, а их реализация — возлагаться на подобного сорта системы. При этом, если ранее использование операторами-шифровальщиками стойких криптоалгоритмов решало все проблемы обеспечения безопасности, то при автоматизированном подходе появлялась необходимость реализовывать процедуры, моделирующие поведение доверенных операторов по доступу к ресурсам. При отсутствии регламента криптоалгоритмы оказываются бесполезны в силу широкой доступности всех ресурсов программам систем. Для его реализации были предложены процедуры идентификации и аутентификации, а также разграничения доступа пользователей. На их основе администраторы системы стали формулировать политику безопасности, обеспечивая необходимое разграничение доступа пользователей к ресурсам.

Однако практическое использование процедур идентификации и аутентификации, а также разграничения доступа показало явную недостаточность этих средств. Во-первых, они не способны в полном объеме реализовать сколько-нибудь сложную политику безопасности. Например, если политика безопасности запрещает легальному пользователю чтение многих доступных ему файлов и агрегацию данных,

Работа частично поддержана грантами Российского фонда фундаментальных исследований 03-01-00248 и 04-07-90010 в 2004 г.

то это требование не реализуемо с применением только вышеперечисленных средств. Во-вторых, в системном программном обеспечении, в том числе и в процедурах операционных систем, которые обеспечивают идентификацию, аутентификацию и логическое разграничение доступа, могут быть как алгоритмические, так и ошибки реализации, приводящие к возникновению уязвимостей. В-третьих, в ходе эксплуатации инфо-телекоммуникационных систем их администраторы и легальные пользователи могут допускать сознательные или ошибочные действия, влекущие за собой нарушение политики безопасности, и, как следствие, появление уязвимости.

Используя уязвимости, существующие в инфо-телекоммуникационных системах, злоумышленники атакуют эти системы с целью нарушения конфиденциальности, целостности или возможности доступа, а также — достижения некоторой произвольной совокупности этих нарушений. Каждая из атак выполняется либо путем подачи некоторой последовательности команд, либо сетевых пакетов. С учетом этого обстоятельства возникает возможность, собирая данные о действиях пользователей и/или системы, выполнять аудит информационной безопасности, в том числе, проводить анализ в динамике изменения состояния системы с целью контроля соответствия поведения пользователей и системы заданной политике безопасности. Подобный динамический аудит называют также мониторингом информационной безопасности.

Следует отметить, мониторинг подразумевает нечто большее, чем функциональность систем обнаружения атак (Intrusion Detect System) или систем предотвращения атак (Intrusion Prevention System). Мониторинг информационной безопасности выполняет сбор, анализ данных с целью выявления в них событий, характеризующих атаки или этапы их подготовки, а также выработку и, может быть, реализацию ответных действий на выявленные атаки. Следует отметить, что ответная реакция на атаку может быть разной в зависимости от исполняемого приложения, так как конечной целью служит исполнение системой ее функций, а не противодействие атакам.

1. Требования к мониторингу информационной безопасности

В качестве основных требований к системам, обеспечивающим мониторинг информационной безопасности могут быть выдвинуты следующие:

- полнота обнаружения атак;
- высокая производительность и масштабируемость;

- минимум ложных тревог;
- умение объяснять причину тревоги;
- интеграция с системой управления и другими сервисами безопасности;
- наличие технической возможности удаленного мониторинга информационной системы.

Полнота обнаружения атак представляет собой очевидное требование. Не столь очевидное требование высокой производительности обусловлено желанием и необходимостью обеспечить условия, при которых система мониторинга сама не порождала бы отказ в обслуживании при возрастании потока событий безопасности, вызванный, в том числе, и увеличением объема (масштабируемость) контролируемых программно-аппаратных средств. Высокая производительность компьютеров и пропускная способность современных каналов связи даже при достаточно малой интенсивности ложных тревог, например 10^{-7}, приводит к тому, что администратор безопасности должен реагировать на десятки атак в час. Собственно эти условия порождают требование необходимости объяснения причины атаки, как средства, помогающего администратору отличить ложную тревогу от реальной атаки. При реализации функций управления и сервисов безопасности необходимо избежать дублирования собираемых данных и производимых над ними предварительных операций обработки, например очистки данных, вычисления средних значений, что требует интеграции этих компонентов. Наконец, организация отказоустойчивости и удаленного управления системой мониторинга обусловливают последнее из вышеперечисленных требований.

2. Сбор данных в системах мониторинга информационной безопасности

В системах мониторинга информационной безопасности используются две основные стратегии (направления) сбора данных:

- сбор данных о поведении аппаратно-программных средств;
- сбор данных о поведении приложений и пользователей.

В случае использования первой стратегии собираются данные, вырабатываемые компонентами инфо-телекоммуникационных систем, такие как значения полей передаваемых пакетов, количество пакетов разных протоколов, последовательности системных вызовов, интенсивности обращений к системным вызовам, поля записей в базах данных, поля в принятых пакетах, векторы прерываний и так далее. При этом из пакетов может выделяться содержимое разных сетевых уровней, произ-

водиться дефрагментация пакетов для получения возможности анализа их полного содержимого и выполняться реконструкция потоков пакетов для получения возможности учета информации о развитии атаки.

При использовании второй стратегии собираемые данные представляют собой последовательности команд, выдаваемых каждым пользователем или приложением, интенсивности выдачи команд и обращений к внешним устройствам для различных временных интервалов, а также другие данные, характеризующие поведение пользователя или приложения.

К собираемым данным предъявляются следующие основные требования:

- полнота, как обеспечение сбора всех значений требуемых данных, например всех системных вызовов без пропусков, возникающих из-за несовершенства применяемого метода и особенностей операционной системы;
- достоверность, как обеспечение неискаженности данных, например, из-за отказов оборудования и действий злоумышленников;
- своевременность, как возможность получения доступа к данным в реальном времени с целью выработки адекватной ответной реакции.

Описание с помощью адекватной математической модели поведения одного из параметров системы, группы параметров или всей системы в целом позволяет аналитически или с помощью численного моделирования дать ответы на вопросы о реальных сроках эксплуатации системы, эффективности настройки ее параметров, надежности, защищенности и ряда других характеристик.

Цифровые телекоммуникационные системы и распределенные вычислительные комплексы в отличие от технических средств предыдущих поколений по своей архитектуре и принципам функционирования могут со сравнительно небольшими дополнительными затратами предоставлять исследователям гигантские объемы информации о происходящих в них процессах. Уже на первых этапах изучения этой информации специалисты стали обращать внимание на то, что получаемые ими данные по своей природе заметно отличаются от тех, с которыми они сталкивались ранее при передаче сигналов, в теории массового обслуживания, электротехнике и других практически значимых направлениях. Работы по этой тематике стали появляться в печати с начала 90-х годов прошлого века. Кратко вычленим основные особенности данных, которые обычно отмечаются в данной предметной области:

- стохастичность большинства изучаемых характеристик, обусловленная самими принципами работы систем, а не помехами и ошибками измерений;
- отсутствие стационарности многих процессов как в «широком смысле» слова (изменяющееся среднее значение и ковариационная функция), так и в «узком» (изменение характера одномерных и многомерных распределений со временем);
- отсутствие гауссовости в распределениях характеристик и возможности описания распределения каким-либо из хорошо известных параметрических семейств распределения вероятностей;
- частое засорение данных нехарактерными значениями;
- самоподобие данных, приводящее к фрактальным процессам.

Так на рис. 1 представлены данные о числе соединений при телекоммуникационном обмене данными. На фоне регулярных внутрисуточных колебаний этого показателя (данные за пять дней) выделяются нехарактерные значения этого показателя, указывающие на возможные атаки.

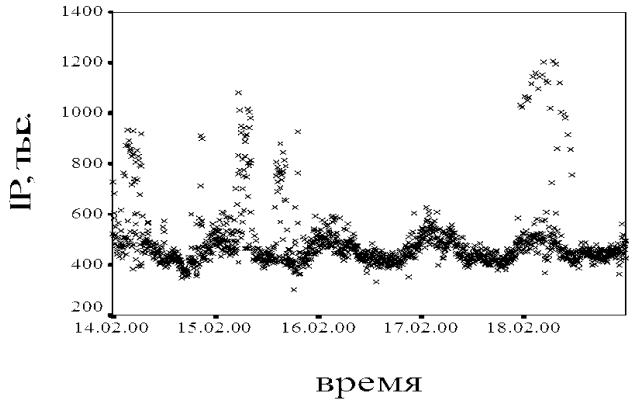


Рис. 1. Нехарактерное число соединений при телекоммуникационном обмене данными на фоне регулярных внутрисуточных колебаний этого показателя.

Указанные особенности данных резко сокращают возможности применения к их анализу традиционных математических алгоритмов, ставят под сомнение целесообразность применения, так называемых, эффективных алгоритмов и алгоритмов, рассчитанных на гауссовский характер распределения случайных отклонений.

Одновременно возникает потребность в изначально более сложных по своей структуре алгоритмов анализа данных даже в сравнительно простых ситуациях. Например, усреднение по времени для стационарного процесса для нестационарного процесса с указанными выше осо-

бенностями приходится заменять однофакторной моделью для локально стационарных участков ряда, а вместо среднего значения (неустойчивого к выбросам) использовать в этой модели медиану.

Кратко основные требования к новым математическим методам и алгоритмам, необходимым для анализа данных мониторинга компьютерных телекоммуникационных систем можно сформулировать следующим образом:

- устойчивость (робастность) к различным, возможным отклонениям данных от исходных предположений;
- свобода от, как правило, неизвестного распределения данных;
- более детальный учет на уровне модели возможной динамики изменения процесса.

Осуществляя мониторинг различных телекоммуникационных и распределенных вычислительных систем, решая различные, актуальные задачи анализа сетевого трафика на протяжении последних десять лет коллективом авторов [1–8] были разработаны и апробированы на практике новые математические методы и алгоритмы анализа данных, отвечающие указанным выше требованиям. В частности были разработаны и реализованы на практике алгоритмы:

- робастной знаковой процедуры оценки и прогноза динамики роста загрузки магистральных каналов телекоммуникационных сетей [1], [3], [4];
- однофакторного знакового оценивания внутрисуточных колебаний скорости передачи данных конечным пользователям [5];
- оценивания сетевой активности пользователей компьютерных сетей [7];
- выделения статистическими методами нехарактерной сетевой деятельности в задачах активного аудита [8].

3. Методы выявления атак

3.1. Классификация методов принятия решений

Процесс принятия решения о том, какие состояния считать атакой относится к классу управленческих и предполагает определение множества признаков объектов, установление шкал и измерение значений этих признаков у объектов, снижение размерности пространства признаков, выявление наиболее информативных признаков, построение решающих правил распознавания классов состояний объектов по векторам признаков, представляющим эти состояния. С учетом изложенного можно утверждать, что решение указанной задачи распознавания служит краеугольным камнем в принятии решений. В дальнейшем вектор

признаков, предъявляемый для распознавания, будем называть входным вектором решающего правила или просто входным вектором.

Методы решения задач распознавания могут быть классифицированы [9] на *лингвистические* (синтаксические, структурные) и *геометрические*.

Лингвистические методы используют в качестве признаков некоторые заранее определенные непроизводные (исходные) элементы, например, поля пакетов, атрибуты базы данных применительно к задачам активного аудита. Состояния объектов представляются посредством иерархической структуры, конструируемой на базе непроизводных элементов. Грамматика задания состояний содержит конечное число непроизводных элементов, правил подстановки и переменных, в лингвистических методах используется весь арсенал формальных языков и грамматик [10]. Лингвистические методы применяются, например, при сигнатурном анализе.

При использовании *геометрических методов* состояния распознаваемых объектов представляются точками в многомерном пространстве признаков, число измерений которого равно числу признаков, различаемых у объектов. Ярким представителем этих методов служат пороговые решающие правила, относящие к разным состояниям, в которых значение некоторого признака больше или меньше заданного порога.

Целью работы, результаты которой изложены далее не является представление всех геометрических методов, поэтому ограничимся только теми из них, которые уже нашли применение в системах мониторинга.

3.2. Полностью определенные методы

При детерминированном геометрическом подходе с заданными функциями выделения классов в качестве таких функций обычно используются либо разделяющие (дискриминантные) функции, либо эталоны [9]. В первом случае границы областей задаются разделяющими функциями, образующими гиперповерхности в пространстве признаков. Например, при линейной отделимости каждого класса используются линейные дискриминантные функции $d(X) = w_1 x_1 + w_2 x_2 + \ldots + w_n x_n + w_{n+1}$, где $X = (x_1, x_2, \ldots, x_n)$ — точка в n-мерном пространстве признаков, $W = (w_1, w_2, \ldots, w_n, w_{n+1})$ — вектор коэффициентов функции $d(X)$.

Для разделения k различных классов A_i, $i = 1, 2, \ldots, k$, задается k дискриминантных функций d_i, $i = 1, 2, \ldots, k$, и решающее правило,

определяемое следующим образом:

$X \in A_i$, если $d_i(X) > 0$, и для всех j, $j \neq i$, $d_j(X) \leqslant 0$, $i, j = 1, \ldots, k$.

Если для какой-либо точки признакового пространства значения нескольких дискриминантных функций больше нуля, то эта точка принадлежит области отказа от принятия решения.

Возможны также попарная и мажоритарная отделимость классов [9], а также использование обобщенных линейных дискриминантных функций вида $d(X) = \sum_{i=1}^{n+1} w_i f_i(X)$, где $f_{n+1}(X) = 1$, а f_i — действительные однозначные функции X. Посредством обобщенных функций возможно задание сложных разделяющих поверхностей.

При отделении классов с помощью одного или нескольких эталонов каждый класс A_i имеет эталон Z_i, $i = 1, \ldots, k$. Используется следующее решающее правило: $X \in A_i$, если $D_i(X) < D_j(X)$ для всех j, $j \neq i$, $i, j = 1, \ldots, k$, где $D_i(X) = \sqrt{\sum_{p=1}^{n}(x_p - z_{ip})^2}$ — расстояние между вектором признаков $X = (x_1, \ldots, x_n)$ и эталоном $Z_i = (z_{i1}, \ldots, z_{in})$.

В случаях областей сложной формы, когда при использовании одного эталона возникает неприемлемая ошибка, прибегают к заданию классов несколькими эталонами. Отнесение входного вектора к определенному классу в этой ситуации выполняется через вычисление расстояний до ближайших эталонов классов.

3.3. Параметрические методы

В отличие от методов с заданными функциями выделения классов, основанных на полной информации, достаточной для построения решающих правил распознавания классов, методы, в основе которых лежит параметрическая неопределенность, требуют предварительной настройки параметров. Значения параметров определяются в ходе, так называемого, обучения, при котором используется совокупность входных векторов, представляющая собой обучающую выборку. Следуя традиции, эти методы называют также нейросетевыми [11].

В нейронных сетях принято следующее представление решаемых задач. Каждому входному параметру x_i, $i = 1, \ldots, n$, задачи сопоставляется измерение i многомерного пространства, размерность которого равна числу n параметров. Для каждого параметра используется некоторая шкала, задающая возможные значения этого параметра. Тем самым постановка задач сводится к определению свойств точек $x_j = \{x_{j1}, x_{j2}, \ldots, x_{jn}\}$ n-мерного пространства, где x_{ji} — значение входного параметра i точки j, при известных свойствах точек, принадлежащих примерам, использованным при обучении.

Итак, пусть имеется обучающий набор примеров

$$\langle X_1, D_1\rangle = \langle (x_{11}, \ldots, x_{1n}), D_1\rangle,$$
$$\langle X_2, D_2\rangle = \langle (x_{21}, \ldots, x_{2n}), D_2\rangle,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$\langle X_m, D_m\rangle = \langle (x_{m1}, \ldots, x_{mn}), D_m\rangle,$$

где $X_j = (x_{j1}, \ldots, x_{jn})$ — входные значения j-го примера, D_j — требуемое выходное значение этого примера, а Y_j — выходное значение, выдаваемое сетью при подаче на ее входы j-го примера, $j = 1, \ldots, m$. Считается, что сеть правильно обучена, если выполняется критерий окончания обучения.

В качестве такого критерия обычно используют следующие условия, хотя могут быть и другие:

- для всех j $\max |D_j - Y_j| \leqslant \delta$, где δ — заданная величина ошибки;
- $\sqrt{\sum_j (D_j - Y_j)^2} \leqslant \delta$.

В задачах, эффективно решаемых на основе нейросетевых подходов, области многомерного пространства, в которых сформулирована задача, образуют множества (совокупности) точек, обладающих одним и тем же свойством. Таким может быть, например, принадлежность одному классу объектов, имеющих одинаковое значение заданной на них некоторой функции и другие подобные критерии. Нейронные сети запоминают подобные области, а не отдельные точки, представляющие предъявленные при обучении примеры.

В ходе функционирования сеть относит предъявленный на ее входы набор значений к той или иной области, что и является искомым результатом. Заметим, что предъявляемый сети набор входных значений мог не подаваться на входы сети при обучении. Однако, в силу сформированных посредством других наборов входных значений совокупности областей, этот набор попадет в одну из них. Если результат правильный, то имеет место правильно функционирующая сеть, иначе сеть обучена или сконструирована с ошибкой. Поэтому смысл процедуры обучения или конструирования — отделение множеств точек, каждой области без включения посторонних точек и потери своих.

Используются различные способы реализации запоминания областей. Наиболее употребляемые в настоящее время это выделение областей посредством гиперплоскостей и покрытие областей гипершарами. Данный подход означает, что используются те же способы, что и в полностью определенных методах решения задач распознавания.

При использовании гиперплоскостей каждый нейрон j с пороговой функцией активации, $j \in \{1, \ldots, N\}$, N — число нейронов в сети, задает

гиперплоскость значениями весов своих входов: $a_j - \sum_{i=1}^{n(j)} w_{ji}x_{ji} = 0$, где $n(j)$ — число входов нейрона j, a_j — величина порога.

Такие сети называются персептронными (одно или многоуровневыми в зависимости от числа слоев) сетями [12].

Изменение числа нейронов, графа межнейронных связей, а также весов входов нейронов меняет количество и положение разделяющих гиперплоскостей, разбивающих многомерное пространство на области. Как показано в [13–15] двухуровневая нейронная сеть способна аппроксимировать в равномерной метрике с любой заданной погрешностью $\varepsilon > 0$ любую непрерывную функцию $f(x_1, x_2, \ldots, x_n)$, а в среднеквадратичной метрике — любую измеримую функцию, определенную на ограниченном множестве:

$$f(x_1, x_2, \ldots, x_n) = \sum_{i=1}^{N} v_i \cdot \frac{1}{1 + \exp(-(w_{i1}x_1 + \ldots + w_{in}x_n))},$$

где v_i — веса входов нейрона второго слоя с линейной функцией активации, w_{ij} — вес j-го входа, $j = 1, \ldots, n$, i-го нейрона, $i = 1, \ldots, N$, первого слоя с сигмоидной функцией активации, N — число нейронов первого слоя.

В случае покрытия гипершарами каждый нейрон задает значениями весов своих входов координаты центра гипершара, а также запоминает радиус этого гиперкуба. Эти сети называются сетями с радиусными базисными функциями.

С учетом изложенного можно утверждать, что в обоих случаях, как при использовании гиперплоскостей и гипершаров, имеет место реализация распределенного коллективного запоминания нейронами при обучении предъявленных сети примеров. При этом объем запоминаемой информации имеет порядок $N(n+1)$, где N — число нейронов сети, а $(n+1)$ — объем информации, необходимой для запоминания гиперплоскости или гипершара в n-мерном пространстве.

Объем запоминаемой информации определяется только необходимостью выделения областей n-мерного пространства, а не числом используемых при обучении примеров. Поэтому сеть может посредством ограниченного набора гиперплоскостей запомнить очень большое число примеров, принадлежащих каждой из выделяемых областей.

4. Пример использования нейросетей для выявления атак

Экспериментальная проверка эффективности применения нейросетей была проведена следующим образом. В качестве данных собирались интенсивности выбранной совокупности системных вызовов. Интенсивность выражается количеством протоколируемых вызовов в секунду (0, если за секунду вызовов не поступило).

В эксперименте использовалась трехслойная сеть прямого распространения (1 скрытый слой). Входной слой содержал 40 нейронов, скрытый слой — 20 нейронов. Сеть обучалась по методу обратного распространения ошибки (BP — Back Propagation) [11].

Обучающая и тестирующая выборки содержали образцы типичного поведения системы в отсутствии атак и образцы, содержащие последовательности системных вызовов, при имитации атак на доступность и атак на переполнение буфера.

Проведенные эксперименты показывают, что точность выявления атак зависит от анализируемого набора системных вызовов, который должен быть адекватен исполняемым приложениям и обнаруживаемым атакам. Так на рис. 2 представлены значения ошибок 1-го и 2-го рода для разных наборов анализируемых системных вызовов при исполнении одного и того же приложения и выявления одних и тех же атак. Лучший результат получен для набора {open, read, fork, exec}, для которого ошибка первого рода составила 0.2%, а ошибка второго рода — 0.8%.

Рис. 2. Процент ошибки при анализе различных наборов системных вызовов.

Тестирование эффективности подсистемы сбора данных показало, что замедление на отработке системных вызовов, порождении и выполнении новых процессов и сценариев оболочки менее 1%. Тестирование эффективности анализатора показало, что система использует менее 1% процессорного времени.

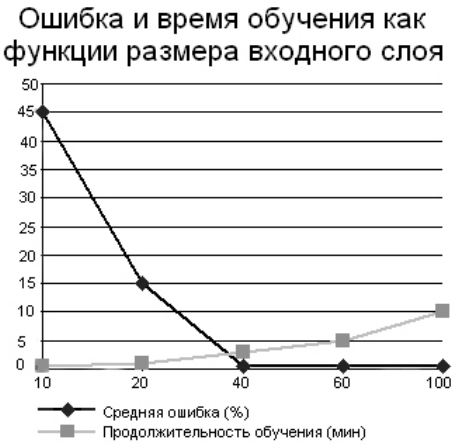


Рис. 3. Зависимости ошибки и времени обучения от количества входов анализирующей нейронной сети.

На рис. 3 приведены зависимости ошибки и времени обучения от количества входов анализирующей нейронной сети, использовавшейся в эксперименте.

Несмотря на то, что существующие нейросетевые парадигмы представляют собой достаточно универсальный аппарат, им свойственен ряд недостатков, основными из которых служат:

- трудность формирования обучающего и тестирующего множеств примеров: при необходимом, вообще говоря, достаточно небольшом множестве примеров для рационального покрытия выделяемых областей, отсутствие информации о покрытии этих областей ведет к тому, что используется излишне большое множество примеров, исходя из одного простого допущения, чем больше примеров, тем больше вероятность покрытия выделяемых областей;
- неприемлемо большое время обучения для достижения требуемой точности функционирования нейросети с большим количеством нейронов, в том числе входных, и большим числом примеров;
- выбор нейронной сети для решения сложной проблемы представляет собой скорее искусство, чем рутинную процедуру и требует высокой квалификации в области нейросетевых алгоритмов.

С целью преодоления этих недостатков развиваются разные подходы, одним из которых служит создание новой нейросетевой парадигмы, основанной на комбинировании нескольких нейросетей.

5. Построение самоорганизующегося иерархического коллектива экспертов

Самоорганизующейся иерархический коллектив экспертов (HBME, Hierarchical Boosted Mixture of Experts) [16] строится с использованием идей древовидной структуры HME [17], алгоритма разбиения обучающего множества [18, 19], а также принципа динамического выбора экспертов. Суть алгоритма обучения HBME состоит в последовательной рекурсивной декомпозиции обучающего множества с целью упрощения обучения экспертов и решателей.

Алгоритм обучения, обеспечивающий формирование иерархической структуры HBME, состоит в следующем. Изначально HBME состоит лишь из одного основного эксперта (basic expert — BE), который обучается на исходном множестве примеров $X = \{x_i, y_i\}$. Критерием окончания обучения может быть выполнение одного из двух условий:

1) достижение требуемой погрешности результатов для всех примеров множества:

$$\forall i : i = 1, 2, \ldots, n; \quad E[x_i, y_i, \widehat{F}(x_i)] < \delta,$$

где $E[x_i, y_i, \widehat{F}(x_i)]$ — значение функционала ошибки эксперта для примера (x_i, y_i), n — число примеров в множестве основного эксперта; δ — заданная погрешность обучения;

2) достижение требуемого соотношения общего количества примеров и числа «правильных» примеров, а именно, — примеров для которых выполняется соотношение $E[x_i, y_i, \widehat{F}(x_i)] < \delta$, после заданного числа эпох обучения $n_{\text{и}}$.

На следующем этапе анализируется количество так называемых «неправильных» примеров, для которых условие $E[x_i, y_i, \widehat{F}(x_i)] < \delta$ не выполняется. Если их число больше некоторого заданного порога $n_{\min}$, то из таких примеров формируется множество для вспомогательного эксперта, и на нем обучается новый экземпляр HBME, реализующий вспомогательного эксперта (auxiliary expert — AE). Если «неправильные» примеры отсутствуют, обучение HBME заканчивается. Число «неправильных» примеров может быть меньше или равно $n_{\min}$. В этом случае их нужно использовать для дообучения основного эксперта.

Алгоритм обучения HBME предусматривает также использование согласующего эксперта (consistent expert — CE). Он создается, чтобы «подстраховывать» основного и вспомогательного экспертов на сложных примерах. Такая необходимость возникает в случае, когда во множествах основного и вспомогательного экспертов по завершении их

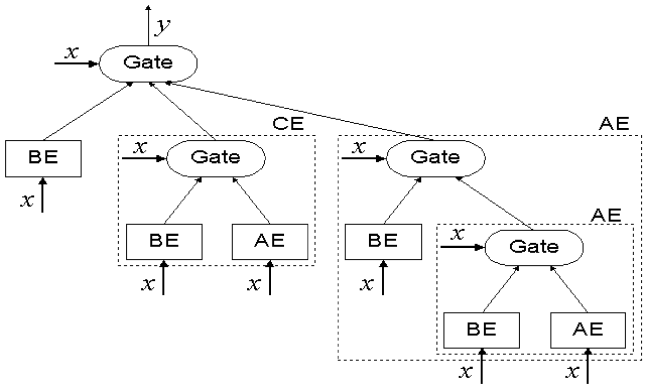


Рис. 4. Структура HBME.

обучения остались «неправильные» примеры. Эти примеры формируют обучающее множество согласующего эксперта, в качестве которого используется новый экземпляр HBME.

Если уровень HBME содержит более одного эксперта, создается решатель (gate), обеспечивающий динамический выбор эксперта в зависимости от вектора аргументов.

Структуру HBME можно представить в виде дерева, листьями которого являются эксперты, а корнями поддеревьев — решатели (рис. 4). На каждом уровне иерархии могут использоваться от одного до трех экспертов. Изначально уровень состоит из одного основного эксперта. В процессе обучения к нему могут добавляться вспомогательный и согласующий эксперты. Динамический выбор эксперта в зависимости от вектора аргументов x реализуется решателем.

Модель HBME была использована для решения задач аппроксимации функций и прогнозирования временных рядов. В использованной реализации модели HBME в качестве экспертов выступали нейронные сети с парадигмой «персептрон с одним скрытым слоем, обучаемый методом обратного распространения ошибки» (BP). В качестве решателей применялись вероятностные нейронные сети. Параметры алгоритма обратного распространения были общими для экспертов HBME и нейросети BP, решающей задачу.

В каждой из задач производилось сравнение результатов ее решения HBME и нейронной сетью с парадигмой BP. Фиксировалось время, необходимое для обучения с заданной погрешностью, также определялась точность получаемых результатов. В задаче аппроксимации

функции оценивалась ошибка отображения на аргументах, которые не были использованы в примерах для обучения. При прогнозировании временных рядов получаемые результаты сравнивались со значениями, действительно наблюдавшимися в интервале, для которого осуществлялось прогнозирование.

Время, затраченное на обучение, вычислялось усреднением значений, полученных на 10 реализациях каждого эксперимента. Для экспериментов был использован IBM-совместимый компьютер с процессором Intel Celeron 400 МГц и 32 Мбайт ОЗУ под управлением ОС Windows 98.

6. Аппроксимация функций

Производилась аппроксимация функций двух переменных $F(x_1, x_2)$, определенных на квадрате $X = (x_1, x_2)$: $x_1 \in [0, 1]$, $x_2 \in [0, 1]$. Для подготовки обучающих примеров использовались значения функции в узлах сетки с шагом $1/20$ по осям x_1 и x_2:

$$\left\{(x_1^{(i)}, x_2^{(k)}) : x_1^{(i)} = \frac{i}{20},\ \ i = \overline{1, 20};\ \ x_2^{(k)} = \frac{k}{20},\ \ k = \overline{1, 20}\right\}.$$

Остановка алгоритма обучения происходила при достижении относительной погрешности результатов, меньшей 0.05, на всем множестве примеров.

Погрешность аппроксимации оценивалась по точкам в узлах сетки

$$\left\{(x_1^{(i)}, x_2^{(k)}) : x_1^{(i)} = 0.25 + \frac{i}{20},\ \ i = \overline{1, 19};\ \ x_2^{(k)} = 0.25 + \frac{k}{20},\ \ k = \overline{1, 19}\right\}.$$

Значения функции F, а также ее аппроксимационные оценки в этих точках, построенные HBME и BP, использовались для расчета среднеквадратического отклонения (СКО), служившего критерием точности аппроксимации.

По результатам экспериментов строились:

- график функции $F(x_1, x_2)$ по точкам сетки (a);
- график оценки функции $F(x_1, x_2)$, полученной с помощью HBME, по точкам сетки (b);
- график оценки функции $F(x_1, x_2)$, полученной с помощью BP, по точкам сетки (c);
- карта разбиения обучающего множества, полученного в результате обучения HBME (закрепление областей входного пространства за экспертами показано цифрами в окружностях) (d).

Ниже представлены графики, а также данные о точности аппроксимации и времени обучения HBME и BP ($t_{\text{обуч}}$) для использованных

в эксперименте функций (1) (рис. 5, 6, 7, 8) и (2) (рис. 9, 10, 11).

$$F(x_1, x_2) = \begin{cases} 1, & (x_1 \in [0.2,\ 0.4] \wedge x_2 \in [0.2,\ 0.4]) \vee \\ & \vee\ (x_1 \in [0.6,\ 0.8] \wedge x_2 \in [0.6,\ 0.8]); \\ 0 & \text{в противном случае.} \end{cases} \quad (1)$$

$$F(x_1, x_2) = \begin{cases} 1, & d(x_1, x_2) < 0.125^2; \\ 0.6, & 0.125^2 \leqslant d(x_1, x_2) < 0.25^2; \\ 0.3, & 0.25^2 \leqslant d(x_1, x_2) < 0.5^2; \\ 0, & 0.5^2 \leqslant d(x_1, x_2). \end{cases} \quad (2)$$

$$d(x_1, x_2) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2.$$

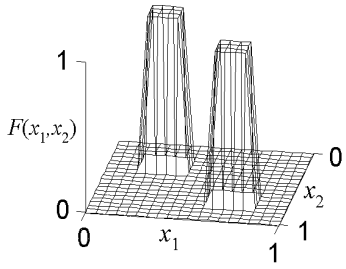


Рис. 5. a)

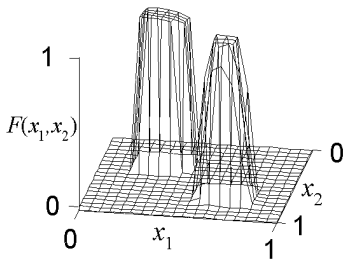


Рис. 6. b) СКО: 0.03, $t_{\text{обуч}}$: 10.8 с.

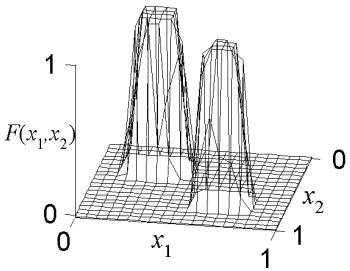


Рис. 7. c) СКО: 0.04, $t_{\text{обуч}}$: 915.9 с.

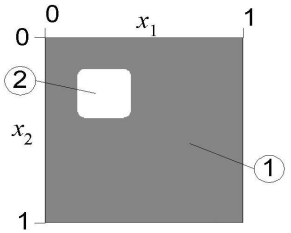


Рис. 8. d)

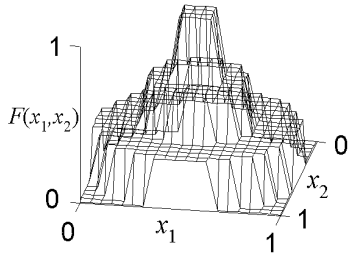


Рис. 9. a)

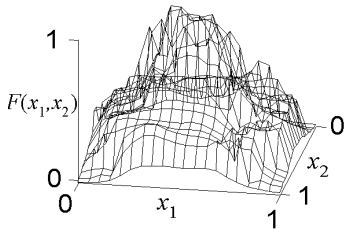


Рис. 10. b) СКО: 0.01, $t_{\text{обуч}}$: 30.9 с.

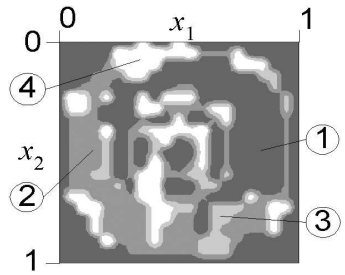


Рис. 11. d)

В эксперименте с функцией (2) не удалось обучить нейронную сеть ВР с требуемой точностью, поэтому график для случая c) отсутствует.

Во всех экспериментах модель НВМЕ обучалась значительно быстрее, чем ВР, при сравнимой точности аппроксимации. Разбиение входного пространства, выполненное НВМЕ в процессе обучения, отражает структуру множества обучающих примеров.

7. Многокомпонентные системы мониторинга

Практика использования различных методов анализа данных в системах мониторинга показывает, ни один из существующих методов не дает приемлемых значений ошибок первого и второго рода, а также то, что различные методы дают на одних и тех же данных разную величину ошибок. Причем метод, дающий наименьшую ошибку, разный для разных данных. Это служит основанием для попытки построения

многокомпонентной системы мониторинга, в которой используется совместная работа нескольких систем мониторинга, применяющих разные методы выявления атак. Идея состоит в том, чтобы использовать для анализа данных именно ту систему мониторинга, которая наилучшим образом подходит для этих данных. Для комбинирования результатов нескольких систем мониторинга в составе многокомпонентной системы мониторинга должен использоваться решатель.

Представляется, что нейросети и, в особенности, НВМЕ могут использоваться не только как аппарат для выявления атак, но и как средство построения решателей, создание которых является основной проблемой при развитии многокомпонентных систем мониторинга.

Заключение

Активный аудит является важным элементом эшелонированного организованного комплекса средств защиты больших практически значимых информационно-вычислительных систем. Использование этого программно-технического сервиса способно во многом устранить недостатки объективно присущие традиционным средствам защиты. Однако эффективная реализация активного аудита сопряжена с необходимостью разработки целого ряда разноплановых математических моделей, их программной реализации на разных уровнях и в составе функционально различных компонент системы. В настоящей работе отмечены некоторые особенности подобных моделей, используемых механизмов и способов анализа данных мониторинга.

Список литературы

[1] *Васенин В. А.* Российские академические сети и Internet. М.: «Московский университет», 1997, 174 с.

[2] *Васенин В. А.*, *Макаров А. А.* Проблемы и методики анализа трафика телекоммуникационных компьютерных сетей. Международная научно-практическая конференция «Новые информационные технологии в университетском образовании», Новосибирск, 25–27 марта 1997 г. с. 173.

[3] *Макаров А. А.*, *Симонова Г. И.* Проблемы робастного оценивания статистических моделей суточных трафиков магистральных каналов компьютерных сетей. Сб. «Статистические методы оценивания и проверки гипотез», Пермский Университет, 1999, вып. 13, с. 171–182.

[4] *Ковба Н. Л.*, *Макаров А. А.*, *Симонова Г. И.* Закономерности изменения загрузки магистральных каналов компьютерных сетей. Автоматика и телемеханика, № 12, 2000, с. 104–114.

[5] *Макаров А. А.*, *Симонова Г. И.* Статистическая модель внутрисуточных колебаний скорости передачи данных пользователям компьютерных сетей. Сб. «Статистические методы оценивания и проверки гипотез», Пермский университет, 2000, вып. 14, с. 144–154.

[6] *Макаров А. А.*, *Симонова Г. И.* Статистические модели динамики роста числа хостов в российских научно-образовательных сетях, Сб.«Статистические методы оценивания и проверки гипотез», Пермский университет, 2001, вып. 15.

[7] *Макаров А. А.*, *Симонова Г. И.*, *Ковба Н. Л.* Проблемы статистического оценивания данных мониторинга в задачах безопасности компьютерных сетей. Сб. «Математические модели в образовании, науке и промышленности», СПб, МАН ВШ, 2003, с. 136–139.

[8] *Васенин В. А.*, *Галатенко А. В.*, *Макаров А. А.* Анализ отдельных компонент трафика в системах активного аудита компьютерных сетей. Материалы конференции «Математика и безопасность информационных технологий» (МГУ, 2003). М.: МЦНМО, 2004, с. 352–364.

[9] *Ту Дж.*, *Гонсалес Р.* Принципы распознавания образов. М.: Мир, 1978.

[10] *Ахо А.*, *Ульман Дж.* Теория синтаксического анализа, перевода и компиляции. М.: Мир, 1978.

[11] *Jain A.*, *Mohiuddin J.* Artificial Neural Networks: A Tutorial. Computer, March, 1996, p. 31–44.

[12] *Hampshir II J.*, *Perlmutter B. A.* Equivalence Proofs for Multy-Layer Perceptron Classifiers and the Bayesian Discriminant Function. Carnegie Mellon University, Pittsburg, 1997.

[13] *Hornick*, *Stinchcombe*, *White.* Multilayer Feedforward Networks are Universal Approximators. Neural Networks, 1989, v. 2, № 5.

[14] *Cybenko.* Approximation by Superpositions of a Sigmoidal Function. Mathematical Control Signals Systems, 1989, 2.

[15] *Funahashi.* On the Approximate Realization of Continuous Mappings by Neural Networks. Neural Networks. 1989, v. 2, № 3.

[16] *Корнеев В. В.*, *Васютин С. В.* Самоорганизующийся иерархический коллектив экспертов. Нейрокомпьютеры: разработка и применение, 2, 2003.

[17] *Jordan M. I.*, *Jacobs R. A.* Hierarchies of adaptive experts. In J. E. Moody, S. J. Hanson and R. P. Lipmann (Eds.). Advances in neural information processing systems. San Mateo, CA: Morgan Kaufmann, 1992, 4, p. 985–992.

[18] *Avnimelech R.*, *Intrator N.* Boosted Mixture of Experts: An Ensemble Learning Scheme. Neural Computation, 1999, 11, p. 483–497.

[19] *Schapire R. E.* The Strength of weak learnability. Machine Learning, 1990, 5(2), p. 197–227.

Security policy languages and enforcement

Scott D. Stoller, Yanhong A. Liu

Introduction

As organizations grow larger and more complex, and as cybersecurity becomes an increasingly important concern, there are growing needs for languages that can express complex security policies of organizations and for efficient mechanisms to enforce the policies. An essential function of security policies is to control *authorization*, that is, to determine whether a request to access a resource should be permitted or denied. This paper considers two key aspects of security policy languages — support for decentralized policy administration through "trust management", and support for scalable policy management through roles — and techniques for the efficient implementation of these languages. The implementation techniques provide a basis for enforcing security policies expressed in these languages.

Traditional ways of expressing authorization policies, such as access control lists, were developed to support authorization in centralized systems with a single administrator in control of all aspects of security policy for the entire system. They are generally adequate and widely used in that context, but they have serious deficiencies for enterprise-wide applications [2], which may have many administrative domains and varying trust relationships.

Trust management systems are designed to support authorization in distributed systems [2]. The defining characteristic of trust management systems is support for decentralized security policy administration through *delegation*: an entity can authorize another entity to control specified aspects of security policy. For example, a company's chief executive officer (CEO) might allow each manager to control his subordinates' access to technical data, while the CEO retains complete control over everyone's access to the company's strategic plan.

Role-based policy languages support scalable specification and management of policies in large systems [11], [4]. A *role* is an abstraction that represents a set of permissions, typically the permissions needed to perform the tasks associated with a position in an organization. Role-based authorization policies specify the roles that each user may adopt, and the permissions associated with each role.

1. Trust management policy languages

Datalog (Database Logic) [5], a classic rule-based query language for databases, is an attractive foundation for policy languages in decentralized systems. Datalog allows recursive definitions of relations, so it can express queries not expressible in relational algebra or relational calculus, but it does not allow construction of recursive data structures, so it consumes bounded resources. Trust management systems based on Datalog or extensions of it include Delegation Logic [7], SD3 (Secure Dynamically Distributed Datalog) [6], Binder [3], and RT (Role-based Trust-management) [8]. Datalog is an attractive basis for trust management languages for several reasons:

1. It is declarative and has a simple and well-studied semantics. Datalog statements can easily be translated into declarative English sentences. This helps ensure that users will be able to formulate security policies that accurately capture their intentions.

2. It allows authorization based on all properties of entities and requests, not only their identities. For example, this enables the above policy involving the CEO, manager, and subordinates to be stated directly and clearly in terms of attributes (e.g., is-a-manager) and relations (e.g., is-the-manager-of), instead of by enumerating the permissions of each person individually.

3. Application-specific relations can be defined, and recursive definitions are allowed. Recursive definitions arise naturally when hierarchical structure (e.g., of a computer network, or of an organization) is involved.

4. Queries are decidable in polynomial time. This includes *compliance checking*, which determines whether a given request is permitted by a given policy, and policy analysis problems, such as computing the *meaning* of a given policy (i.e., the set of all operations permitted by the policy).

A significant obstacle to deployment of trust management systems with Datalog-based policy languages is the lack of suitable implementations of such languages. Simple Datalog interpreters are easy to implement but have poor performance, especially for programs containing recursive definitions. Development of an optimized implementation is a significant undertaking, and the result is a heavy-weight system, not easily deployed for trust management [8]. Worse yet, even in the best existing highly-optimized logic programming systems, such as GNU Prolog and XSB (`http://xsb.sourceforge.net/`), the running time of a program can vary dramatically depending on the order of rules in the program or the

order of hypotheses within each rule. It is very difficult for users to determine which order will lead to more efficient execution, because the answer depends on implementation details.

We are addressing these implementation challenges through development of a powerful method, described in [9], for (1) automatic transformation of specifications in Datalog-like languages into specialized algorithms and lightweight implementations, and (2) automated complexity analysis that provides time and space guarantees for the generated algorithms.

The transformation is based on a general method for efficient fixed-point computation. The set of all facts derivable from a Datalog program corresponds to a fixed-point computation that repeatedly derives new facts from existing facts by using the rules, until no more new facts can be derived. Our transformation exploits three key ideas to compute the fixed point efficiently: (1) perform a minimum update in each iteration, i.e., add one new fact at a time; (2) maintain appropriate auxiliary maps (i.e., indices), based on the structure of the rules, and update them incrementally in each iteration; and (3) use appropriate combinations of indexed and linked data structures.

The generated implementations have guaranteed optimal time complexity and associated space complexity, in the sense that only useful combinations of information that lead to the invocation of a policy rule are considered, and each such combination is considered exactly once. The method computes the worst-case time and space complexities, as formulas. These are independent of the order of rules and hypotheses, in contrast to previous methods.

We are extending the method to handle query-driven (goal-directed) computation, incremental computation with respect to policy changes, and distributed evaluation.

2. Role-based policy languages

Role-based policy languages offer well-established advantages for policy management in large systems. They are widely used in role-based access control (RBAC) [11]. The roots of role-based access control include the notion of groups in UNIX and other operating systems. The growing importance of RBAC motivated the development and recent approval of an ANSI standard for it [4], [1].

The standard specifies sets of users, objects, operations, roles, and sessions, and over a dozen relationships built on top of these sets. The standard then specifies several dozen operations on them.

Producing straightforward implementation of the standard in a high-level language with good support for sets, such as the popular object-oriented scripting language Python (`http://www.python.org/`), is relatively easy, but the straightforward implementation will have poor performance.

Manual development of a high-performance implementation takes significantly longer and requires writing more complicated and less modular code, which consequently is also more difficult to maintain. To see why, note that it is often much more efficient to incrementally maintain a set (or a relation, which we regard as a set of tuples) than to compute it from scratch each time it is needed. In a straightforward implementation, a set used in one operation may be calculated from scratch in one place, in the code for that operation. To produce an efficient implementation, we must identify all operations that perform updates that may affect that set, and in their implementations insert code to incrementally maintain that set. This breaks the modularity (abstraction) of the straightforward design and implementation. More generally, this problem arises with all expensive computed quantities (not only sets), which we refer to as *queries* of the data from which they are computed.

We are addressing this implementation challenge through development of a general and systematic method that supports the use of abstractions by allowing each component and operation to be specified in a clear and modular fashion and implemented straightforwardly in an object-oriented language [10]. The method then analyzes queries and updates in the straightforward implementation, cutting across the abstractions in the clear and modular specification. Finally, the method breaks through the abstractions and transforms the straightforward implementation into a sophisticated and efficient implementation that incrementally maintains the results of expensive queries with respect to all relevant updates. The main steps are to determine which queries should be incrementally maintained, which updates may affect each query, and, most challengingly, where to store and how to update each incrementally maintained value, based on a cost model. The transformations are expressed declaratively as *incrementalization rules*.

The method can be used automatically, semi-automatically, or manually. We developed conservative analysis and transformations that can be fully automated. A semi-automatic version may use more aggressive analysis and transformations that rely on hints from the user. The method can be followed manually as a design methodology.

We developed a prototype implementation of the method for Python. We applied the method in the development of efficient object-oriented programs in a number of example applications [10]. Here we discuss the application to RBAC.

We created a straightforward implementation of the RBAC specification in Python, as a complete and executable specification. Overall, more than

half of the operations involve expensive queries, i.e., queries that take more than constant time.

We then applied our analyses and transformations to our straightforward implementation of core RBAC (core RBAC contains the majority of the sets, relations, and operations in the RBAC standard), to automatically incrementalize the expensive queries with respect to the updates. There are over a dozen expensive queries and over a dozen updates. Our method is able to optimize all expensive queries to take constant time except for a tradeoff that lets either `CreateSession` take constant time and `CheckAccess` (and `SessionPermissions`) take $O(|\texttt{Permissions}|)$ time, or *vice versa*. In typical applications, `CheckAccess` is performed much more frequently than `CreateSession`, and thus the latter gives better performance, as confirmed in our experiments. For example, with 900 permissions, the average running time of `CheckAccess`, for the straightforward, the former incrementalized, and the latter incrementalized implementations are 0.342, 0.0345, and 0.000187 seconds, respectively, when measured on a dual-CPU 2.3 GHz Athlon XP computer with Python 2.3.

Future research is needed on improved analysis of costs and frequencies of and dependencies between queries and updates, suitable languages for specifying incrementalization rules, and further optimizations for on-demand and concurrent computations.

We are also investigating the design and implementation of role-based trust management languages.

References

[1] American National Standards Institute, Inc. Role-based access control ANSI INCITS 359-2004. `http://csrc.nist.gov/rbac/`.

[2] *Matt Blaze, Joan Feigenbaum, John Ioannidis, Angelos D. Keromytis*. The role of trust management in distributed systems. In Secure Internet Programming, volume 1603 of Lecture Notes in Computer Science, pages 185–210. Springer-Verlag, 1999.

[3] *John DeTreville*. Binder, a logic-based security language. In Proc. 2002 IEEE Symposium on Security and Privacy, pages 105–113. IEEE Computer Society Press, 2002.

[4] *David F. Ferraiolo, Ravi Sandhu, Serban Gavrila, D. Richard Kuhn, Ramaswamy Chandramouli*. Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security*, 4(3): 224–274, 2001.

[5] *Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer Widom*. Database Systems: The Complete Book. Prentice-Hall, 2002.

[6] *Trevor Jim*. SD3: A trust management system with certified evaluation. In *Proc. 2001 IEEE Symposium on Security and Privacy*, pages 106–115. IEEE Computer Society Press, 2001.

[7] *Ninghui Li, Benjamin N. Grosof, Joan Feigenbaum*. Delegation logic: A logic-based approach to distributed authorization. ACM Transaction on Information and System Security, 2003.

[8] *Ninghui Li, John C. Mitchell, William H. Winsborough*. Design of a role-based trust management framework. In Proc. 2002 IEEE Symposium on Security and Privacy, pages 114–130. IEEE Computer Society Press, 2002.

[9] *Yanhong A. Liu, Scott D. Stoller*. From Datalog rules to efficient programs with time and space guarantees. In Proceedings of the 5th ACM SIGPLAN International Conference on Principles and Practice of Declarative Programming, pages 172–183, Uppsala, Sweden, August 2003.

[10] *Yanhong A. Liu, Scott D. Stoller, Yanni Ellen Liu, Michael Gorbovitski, Tom Rothamel*. Design by building up and breaking through abstractions. Technical Report DAR-04-15, SUNY at Stony Brook, Computer Science Dept., September 2004.

[11] *Ravi Sandhu, Edward Coyne, Hal Feinstein, Charles Youman*. Role-based access control models. IEEE Computer, 29(2): 38–47, 1996.

Секция «Математические проблемы информационной безопасности»

О поточных шифраторах с динамически изменяющимся законом шифрования

В. С. Анашин

В качестве математической модели поточного шифратора обычно рассматривают автомат с внутренним алфавитом A и выходным алфавитом B, последовательность внутренних состояний (промежуточная гамма) которого определяется законом $x_{i+1} = f(x_i)$, а выходная последовательность (гамма наложения, или выходная гамма) — законом $y_i = F(x_i)$, где $i = 0, 1, 2, \ldots$, $f\colon A \to A$, $F\colon A \to B$ — некоторые отображения. Если $|A| = K$, то отождествим A с кольцом вычетов $\mathbb{Z}/K$. В данном сообщении мы ограничиваемся случаем $K = 2^n$, $A = B$.

Чаще всего и закон рекурсии f, и функция усложнения F не зависят от ключа, которым является начальное состояние x_0, т. е. считаются известными противнику. Стойкость такой схемы обеспечивается, среди прочего, большой трудоемкостью решения системы уравнений $y_i = F(x_i)$ относительно x_i при известных противнику y_i. Поскольку нахождение x_{i-1} из уравнения $x_i = f(x_{i-1})$ при известных f и x_{i-1} обычно не вызывает сложностей, то основной вклад в обеспечение стойкости вносит функция F. В связи со сказанным криптографами предпринимались попытки максимально усложнить противнику задачу решения уравнения $y_i = F(x_i)$: предлагались, например, криптосхемы, где F зависит от ключа, или даже меняется от такта к такту. Помимо этого, рассматривались и криптосхемы, для которых трудоемкость нахождения x_{i-1} по x_i велика (последнее также может достигаться за счет того, что f зависит от ключа или меняется от такта к такту) — см. [1, Ch. 16].

Криптосхемы такого рода имеют общий недостаток, присущий всем криптосхемам «дикого» типа, а именно — отсутствие теоретического обоснования значений критичных для криптографии характеристик выходной последовательности, таких как длина периода, линейная сложность, распределение комбинаций и т. п. Мы пользуемся здесь предложенной Климовым и Шамиром терминологией: согласно [4], «ручные» криптосхемы строятся из простых примитивов (типа линейных регистров сдвига), чье поведение хорошо изучено, с тем чтобы математически доказать определенные криптографически значимые свойства, в то

время как в «диких» используются «безумные» (crazy, так в подлиннике!) комбинации различных операций в надежде на то, что ни разработчик, ни противник не сумеют математически описать поведение криптосхемы и тем самым снизить ее стойкость по сравнению с полным перебором всех ключей. В той же работе [4] авторы отмечают, что «ручные» схемы в основном предпочитаемы в учебниках или в качестве примеров, в то время как реальная практика чаще ориентируется на «дикий» тип.

Целью данного сообщения является изложение некоторого общего подхода к построению поточных шифраторов «полудикого» (в терминологии все той же работы [4]) типа, т. е. таких, в которых и закон рекурсии, и функция усложнения могут зависеть от ключа и меняться от такта к такту, но которые, тем не менее, генерируют выходную последовательность, имеющую математически обоснованные характеристики — максимальную длину периода, большой линейный ранг, равномерное распределение k-грамм. Внешне эти функции действительно могут выглядеть как «безумные»; например[1], функция перехода может быть задана в виде $f(x)=1+x+2\cdot(g(x+1)-g(x))$, где

$$g(x)=\left(1+2\cdot\frac{x\ \mathsf{AND}\ x^2+x^3\ \mathsf{OR}\ x^4}{3+4\cdot(5+6x^5)^{x^6\ \mathsf{XOR}\ x^7}}\right)^{7+8x^8/(9+10x^9)}.$$

В этом случае промежуточная гамма, образованная по закону $x_{i+1}\equiv f(x_i) \pmod{2^n}$, представляет собой бинарную последовательность периода точно $n\cdot 2^n$, которая строго равномерно распределена, т. е. каждая k-грамма ($k=1, 2, \ldots, n$) встречается на периоде одинаковое число раз. Линейная сложность этой последовательности не менее 2^{n-1}. Кроме того, эта последовательность «случайна по Кнуту»: ее период удовлетворяет его условиию Q1 случайности конечной последовательности.

Для удобства читателя напомним, что согласно [13, Section 3.5, Definition Q1] конечная бинарная последовательность $\varepsilon_0\varepsilon_1\ldots\varepsilon_{N-1}$ длины N считается случайной, если

$$\left|\frac{\nu(\beta_0\ldots\beta_{k-1})}{N}-\frac{1}{2^k}\right|\leqslant\frac{1}{\sqrt{N}} \tag{1}$$

для всех $0<k\leqslant\log_2 N$, где $\nu(\beta_0\ldots\beta_{k-1})$ число всех вхождений бинарного подслова $\beta_0\ldots\beta_{k-1}$ в бинарное слово $\varepsilon_0\varepsilon_1\ldots\varepsilon_{N-1}$. Отметим, что в отличие от строгой равномерной k-распределенности, которая влечет строгую равномерную $(k-1)$-распределенность, чтобы доказать, что

[1] Мы приводим этот пример лишь для иллюстрации возможной степени безумия, а не для практической реализации.

конечная последовательность длины N удовлетворяет Q1, недостаточно показать, что (1) выполняется для $k=\lfloor\log_2 N\rfloor$. Например, последовательность 1111111100000111 удовлетворяет (1) для $k=\lfloor\log_2 n\rfloor=4$, но не удовлетворяет (1) для $k=3$.

Оказывается, что все вышеперечисленные свойства последовательности $\{x_i\}$ сохраняются, если в качестве g взять *любую* композицию арифметических операций — сложения $(y, z)\mapsto y+z$, умножения $(y, z)\mapsto yz$, экспоненцирования $(y, z)\mapsto(1+2y)^z$ (в частности, взятия обратного элемента, $y\mapsto(1+2y)^{-1}$), поразрядных логических операций — конъюнкции $(y, z)\mapsto y\ \mathsf{AND}\ z$, дизъюнкции $(y, z)\mapsto y\ \mathsf{OR}\ z$, исключительного «или» $(y, z)\mapsto y\ \mathsf{XOR}\ z$, отрицания $z\mapsto\mathsf{NOT}\ z$, и «машинных» операций (которые, в сущности, являются производными операциями от перечисленных выше) — сдвига на s шагов в сторону старших значащих разрядов $z\mapsto 2^s z$, маскирования $z\mapsto z\ \mathsf{AND}\ M$ с маской M, приведения по модулю 2^s, состоящая в отбрасывании всех старших значащих бит $z\mapsto z \bmod 2^s = z\ \mathsf{AND}\ (2^s-1)$,[2] и некоторых других. При этом предполагается, что все вышеуказанные операции осуществляются над элементами кольца $\mathbb{Z}/2^k$, причем логические операции производятся поразрядно над операндами, представленными в двоичной системе счисления: так, $2=1\ \mathsf{XOR}\ 3=2\ \mathsf{AND}\ 7\equiv\mathsf{NOT}\ 13 \pmod 8$, $3^{-1}\equiv 11\equiv -5 \pmod{16}$, $3^{-1/3}\equiv 3^{11}\equiv 3^{-5}\equiv 11 \pmod{16}$ и т. п. При таких соглашениях функции g и f определены на $\mathbb{Z}/2^n$ корректно, а сложность их программной реализации определяется исключительно соотношением и количеством «быстрых» и «медленных» операций в композиции g, т. е. может произвольно варьироваться в зависимости от требований к быстродействию.

Кроме того, функции f и F могут зависеть от номера такта, т. е. закон образования выходной последовательности будет иметь вид

$$\begin{cases} x_{i+1}\equiv f_{i \bmod m}(x_i) \pmod{2^n};\\ y_i\equiv F_{i \bmod m}(x_i) \pmod{2^n}. \end{cases}$$

Иными словами, в память машины в *произвольном порядке*[3] записываются m функций f_i и m функций F_i, вообще говоря, зависящих от ключа, т. е. неизвестных противнику[4], которые поочередно извлека-

[2] Заметим, что ассемблер практически любого современного процессора содержит сложение, умножение и вышеуказанные поразрядные логические и машинные операции.

[3] Который тоже может определяться ключом.

[4] Он знает лишь их общий вид: например, что $f_i(x)=1+x+2\cdot(g_i(x+1)-g_i(x))$, где $g_i(x)$ какая-то, *неизвестная* ему композиция упомянутых выше арифметических, логических и машинных операций.

ются из памяти для образования очередной n-граммы выходной гаммы по вышеуказанному закону.

Оказывается,что даже при довольно слабых ограничениях на f_i и F_i можно гарантировать, что выходная гамма имеет максимальный период $m \cdot 2^n$, строгое равномерное распределение и линейный ранг не менее $2^{n-1}+1$. Например, эти свойства имеют место при выполнении следующих условий.

Пусть $m \equiv 3 \pmod 4$ — положительное целое число, $v_0(x)$, …, $v_{m-1}(x)$ и $w_0(x)$, …, $w_{m-1}(x)$ — *произвольные* композициии вышеупомянутых арифметических, логических и машинных операций. Возьмем в качестве начального состояния произвольное $x_0 \in \{0, 1, \ldots, 2^n-1\}$ и положим

$$\begin{cases} x_{i+1} = (i \bmod m + x_i + 4 \cdot v_{i \bmod m}(x_i)) \mod 2^n; \\ z_i = (1 + \pi(x_i) + 4 \cdot w_{i \bmod m}(\pi(x_i))) \mod 2^n, \end{cases}$$

где $i \bmod m$ — наименьший неотрицательный вычет i по модулю m, а π — перестановка, изменяющая порядок следования разрядов n-разрядного числа $z \in \{0, 1, \ldots, 2^n-1\}$ на обратный: так, $\pi(0)=0$, $\pi(1)=2^{n-1}$, $\pi(2)=2^{n-2}$, $\pi(3)=2^{n-2}+2^{n-1}$ и т. д. Тогда длина периода последовательности $\{x_i\}$ n-битных слов (рассматриваемых как числа из $\{0, 1, \ldots, 2^n-1\}$) равно точно $m \cdot 2^n$, и каждое такое число встречается на периоде точно m раз. Более того, если рассмотреть последовательность $\{x_i\}$ как бинарную последовательность (имеющую, следовательно, период $2^n mn$), то тогда частота встречаемости каждой k-граммы $(0 < k \leqslant n)$ в этой последовательности в точности равна $1/2^k$. Длина периода выходной последвательности $\{z_i\}$ также равна $m \cdot 2^n$, причем каждый элемент из $\{0, 1, \ldots, 2^n-1\}$ встречается на периоде в точности m раз. Наконец,длина периода каждой бинарной подпоследовательности $\{\delta_s(z_i)\colon i = 0, 1, 2, \ldots\}$, составленной из всех s-ых бит $\delta_s(z_i)$ $(0 \leqslant s \leqslant n-1)$ выходной последовательности, делится на 2^n (а значит, не менее 2^n), а ее линейный ранг (а значит, и линейный ранг всей последовательности $\{z_i\}$) превосходит 2^{n-1}.

Еще одно замечание касается программной реализации вышеупомянутых алгоритмов. Все они используют операции над n-разрядными числами, поэтому, формально говоря, требуют n-разрядного процессора[5]. Это может оказаться неудобным при больших n (скажем, при $n > 64$); кроме того, обычно все современные программные криптоалгоритмы ориентируются на 32-разрядные машины, но период длины 2^{32}

[5]Или программной реализации соответствующих операций над очень большими числами, что сильно снижает быстродействие.

для современных поточных шифраторов считается недостаточным. Тем не менее, как показано в [6], хотя все приведенные выше преобразования формально и являются одномерными преобразованиями кольца вычетов $\mathbb{Z}/2^n$, их можно представить как r-мерные преобразования кольца меньшего порядка $\mathbb{Z}/2^k$, где $n = rk$, которые являются композициями вышеупомянутых арифметических, логических и машинных операций разрядности k. Таким образом, закон рекурсии будет иметь вид

$$\begin{cases} \mathbf{x}_{i+1} = \mathbf{f}_{i \bmod m}(\mathbf{x}_i); \\ \mathbf{y}_i = \mathbf{F}_{i \bmod m}(\mathbf{x}_i), \end{cases}$$

где

$$\mathbf{x} = \left(x^{(1)}, \ldots, x^{(r)}\right) \in \left(\mathbb{Z}/2^k\right)^r,$$

$$\mathbf{f}_j(\mathbf{x}) = \left(f_j^{(1)}(\mathbf{x}), \ldots, f_j^{(r)}(\mathbf{x})\right),$$
$$\mathbf{F}_j(\mathbf{x}) = \left(F_j^{(1)}(\mathbf{x}), \ldots, F_j^{(r)}(\mathbf{x})\right),$$

а $f_j^{(\cdot)}$, $F_j^{(\cdot)}\colon \mathbb{Z}/2^k \to \mathbb{Z}/2^k$ — некоторые композиции упомянутых выше операций разрядности k. Из сказанного следует, что, например, при $n = 256$ нужно будет работать с 8-мерными массивами 32-разрядных слов; при этом выходная бинарная гамма будет иметь период длины $m \cdot 2^{264}$, строгое равномерное распределение всех s-грамм до $s = 256$ включительно и линейную сложность не менее 2^{255}.

За недостатком места мы не имеем возможности привести здесь точные формулировки соответствующих теорем и отсылаем читателя к препринтам [5] и [6]. Отметим лишь, что в них, а также в статьях [7–10] разработан математический аппарат, позволяющий строить широкие классы автоматов с указанными выше свойствами. Все они строятся на основе эргодических преобразований пространства целых 2-адических чисел, при этом сами автоматы (т. е. поточные шифраторы) представляют собой динамические системы (автономные или неавтономные) на этом пространстве. Изначально такой подход был предложен автором в [9], [8]. Позднее идея построения поточных шифраторов на основе преобразований такого типа начала развиваться также Климовым и Шамиром в [2–4][6]. Идея изменять от такта к такту функцию усложнения в псевдослучайных генераторах так, чтобы тем не менее гарантировать отсутствие коротких циклов в выходной последовательности, обсуждалась Шамиром и Тсабаном в [11]. Как отмечено в [12], предложенная там композиция автоматов представляет собой

[6]Как показано в [5] и [6], основные результаты этих работ несложно получаются с помощью развитой автором техники.

эргодическое косое произведение (хотя на деле и сводится, в сущности, к «подмешиванию» счетчиковой последовательности в генератор с неизвестной цикловой структурой).

В заключение отметим, что проведенные машинные эксперименты показали высокое быстродействие криптосхем, построенных на основе изложенного подхода. Кроме того, выработанные ими последовательности успешно прошли все статистические тесты NIST и DIEHARD.

Список литературы

[1] *Schneier B.* Applied Cryptography. John Wiley and Sons, 1996.

[2] *Klimov A.*, *Shamir A.* A new class of invertible mappings. In. Cryptographic Hardware and Embedded Systems 2002 (B. S. Kaliski Jr. et al., eds.), Lect. Notes in Comp. Sci., Vol. 2523, Springer-Verlag, 2003, pp. 470–483.

[3] *Klimov A.*, *Shamir A.* Cryptographic applications of T-functions. In: Selected Areas in Cryptography, 2003.

[4] *Klimov A.*, *Shamir A.* New Cryptographic Primitives Based on Multiword T-functions. 2004 (to appear).

[5] *Anashin V.* Pseudorandom Number Generation by p-adic Ergodic Transformations. 2004, `http://arXiv.org/abs/cs.CR/0401030`.

[6] *Anashin V.* Pseudorandom Number Generation by p-adic Ergodic Transformations: an Addendum. 2004, `http://arXiv.org/abs/cs.CR/0402060`.

[7] *Анашин В. С.* Равномерно распределенные последовательности целых p-адических чисел, II. Дискретная математика, **14** (2002), № 4, стр. 3–64. Препринт на английском языке доступен на `http://arXiv.org/abs/math.NT/0209407`.

[8] *Anashin V.* Uniformly distributed sequences over p-adic integers. Number theoretic and algebraic methods in computer science. Proceedings of the Int'l Conference (Moscow, June–July, 1993, A. J. van der Poorten, I. Shparlinsky, H. G. Zimmer, eds.), World Scientific, 1995, 1–18.

[9] *Анашин В. С.* Равномерно распределенные последовательности целых p-адических чисел. Мат. заметки, **55** (1994), № 2, стр. 3–46.

[10] *Anashin V. S.* Uniformly distributed sequences in computer algebra, or how to construct program generators of random numbers. J. Math. Sci. (Plenum Publishing Corp., New York), **89** (1998), No. 4, pp. 1355–1390.

[11] *Shamir A.*, *Tsaban B.* Guaranteeing the diversity of number generators. Information and Computation, **171** (2001), pp. 350–363. Available at `http://arXiv.org/abs/cs.CR/0112014`.

[12] *Tsaban B.*, 2004 (частное сообщение).

[13] *Knuth D.* The Art of Computer Programming. Vol. 2: Seminumerical Algorithms. Third edition. Addison-Wesley, Reading, MA, 1998.

Об алгебраической иммунности рекурсивных конструкций нелинейных фильтров

А. А. Ботев

Потоковый шифратор — довольно распространенный криптографический инструмент, применяющийся для шифрования сообщений. Как правило, потоковые шифраторы состоят из линейной части, порождающей последовательность с большим периодом, обычно состоящей из одного или нескольких регистров сдвига с линейной обратной связью (РСЛОС), и нелинейной комбинирующей функции f, которая порождает выходную последовательность по данным линейным входам. Линейная часть зависит от «ключа», или, другими словами, начальных значений ячеек памяти. Исследования криптографической устойчивости таких шифраторов большей частью сводятся к исследованию нелинейной функции f, в частности, к исследованию этой функции с точки зрения того, удовлетворяет или не удовлетворяет она некоторым критериям, необходимым для того, чтоб успешно противостоять различным криптографическим атакам.

Среди криптографических атак, позволяющих злоумышленнику узнать «ключ» и в дальнейшем читать все наши сообщения, наиболее распространены корреляционные, линейные и другие. Поэтому существует ряд критериев, которым должны удовлетворять фильтры, чтобы противостоять этим атакам. Наиболее важными свойствами являются уравновешенность, высокие нелинейность, алгебраическая степень, корреляционная иммунность. Все эти свойства сложно взаимодействуют друг с другом, зачастую конфликтуя между собой. Например, известное неравенство Зигенталера утверждает, что корреляционно-иммунная порядка m функции от n переменных не может иметь алгебраическую степень большую, чем $n-m-1$. Поэтому функций, оптимальных по всем параметрам одновременно, не существует. Кроме того, время от времени появляются новые атаки, накладывающие новые ограничения на комбинирующую РСЛОС функцию.

В 2002–2003 годах французским исследователем Николасом Куртуа (N. Courtois) и рядом его коллег был разработан новый вид криптографической атаки — так называемая алгебраическая атака [5], [6],

оказавшаяся весьма эффективной против рассматриваемого нами здесь типа поточных шифраторов. Данная атака основана на линеаризации точной или приближенной системы нелинейных уравнений невысокой степени, связывающей начальные значения ячеек памяти РСЛОС и выходные биты РСЛОС. В связи с разработкой этой атаки появились новые требования надежности, которые выдвинул сам Н. Куртуа. Булева функция f, используемая в качестве нелинейного фильтра не только не должна иметь хорошей аппроксимации функциями невысокой степени (то есть быть на небольшом расстоянии до кодов Рида — Мюллера), но и не должно существовать функции g невысокой степени, такой что функция $f \cdot g$ также невысокой степени или хотя бы хорошо аппроксимируется функцией невысокой степени. Н. Куртуа показал, что против алгебраической атаки любой шифратор с комбинирующей функцией, у которой не более чем десять входов, заведомо не удовлетворяет требуемым критериям надежности. Для большинства известных алгебраических конструкций фильтров с большим числом входов также были предложены алгебраические атаки, намного более эффективные, чем простой перебор ключей.

В недавних работах [3], [4] было показано, что для того, чтобы успешно противостоять алгебраическим атакам, функция должна обладать хорошей алгебраической иммунностью.

Впервые понятие алгебраической иммунности появилось в работе [3]. *Алгебраической иммунностью* функции f, или $\mathrm{AI}(f)$ называется минимальное натуральное число k такое, что существует функция g степени k, обнуляющая f или $f+1$ (т. е. что $f \cdot g = 0$ или $(f+1) \cdot g = 0$).

В работе [6] приведена верхняя оценка на степень алгебраической иммунности произвольной функции от n переменных — а именно, показано, что алгебраическая иммунность такой функции не может превышать $[n/2]$. В работе же [3] приведена нижняя оценка — правда, мощностная, — т. е. показано, при каком отношении d к n вероятность того, что некоторая функция степени не больше d является аннигилятором для уравновешенной булевой функции от n переменных, стремится к нулю при $n \to \infty$. Для справки — это отношение равно примерно 0.27.

Отметим, что этот подход не является практически приемлемым — хотя для каждой уравновешенной функции можно оценить сверху вероятность того, что существует функция степени не больше чем d, являющаяся ее аннигилятором, для каждой конкретной функции оценить сверху порядок ее алгебраической иммунности невозможно. Более того, во всех предыдущих работах, относящихся к этой теме, вообще не было указано никаких нижних оценок алгебраической иммунности для конкретных функций.

В работе докладчика рассматривается одно из семейств фильтров на основе алгебраических конструкций, разработанное и развитое Ю. В. Таранниковым в работах [1], [7–10]. Фильтры, задаваемые этими конструкциями, являются оптимальными во многих смыслах, в частности, для широких границ параметров были достигнуты теоретически оптимальные соотношения между нелинейностью и корреляционной иммунностью, нелинейностью и алгебраической степенью каждого входа, линейной сложностью построения и т. д. Схема поточного шифратора, состоящего из РСЛОС и нелинейного фильтра, построенного на основе этой конструкции, приведена в прошлогодней работе [2].

Сущность данной конструкции состоит в том, что из двух функций f_0 и f_1 с некоторыми заданными криптографическими свойствами можно путем прибавления линейных переменных, переобозначения переменных и конкатенации получить две следующие функции, f'_0 и f'_1, с теми же самыми криптографическими свойствами, из которых, в свою очередь можно построить следующие две, и так далее, до $f_0^{(k)}$ и $f_1^{(k)}$ для любого k. Была доказана следующая теорема:

Теорема 1. *Обозначим через F набор функций f_0 и f_1, и F' — набор функций f'_0 и f'_1. Тогда для любых $h \in F$ и $h' \in F'$ справедливо равенство*

$$\mathrm{AI}(h') \geqslant \mathrm{AI}(h).$$

Это значит, что если ни некоторую функцию, ни ее отрицание нельзя обнулить многочленом степени меньше, чем k, то и все последующие функции (и их отрицания) также нельзя обнулить многочленами степени меньше, чем k. Часто для функции с большим числом входов искать ее алгебраическую иммунность нецелесообразно, достаточно проверить, больше ли она, чем некоторое d. В таком случае можно начать проверку с функций, зависящих от не слишком большого n, и, если ее алгебраическая иммунность выше, чем d, дальнейшую проверку не производить. В работе [3] указана вычислительная сложность и требующаяся память для наилучшего на данный момент алгоритма проверки, имеет ли булева функция от n переменных аннигилятор степени d, равные соответственно $\frac{1}{8}\binom{n}{d}^3$ и $\frac{1}{4}\binom{n}{d} \cdot \binom{n}{d-1}$. При проверке функции, у которой не n, а $n-3k$ входов (напомним, что в рассматриваемой рекурсивной конструкции после каждой итерации количество входов увеличивается как минимум на три), вычислительная сложность уменьшается более чем в $(1+d/(n-d))^{9k}$ раз, а требующаяся память — в $(1+(d-1)/(n-d))^{6k}$ раз, что особенно эффективно при d, значительных относительно n.

Более того, для функций, рассмотренных в [8] доказана

Теорема 2. *В обозначениях предыдущей теоремы, если* $\mathrm{AI}(h) = d$, *степень* $\deg(h) = k$, *то*

$$\mathrm{AI}\left(h^{(k-1)}\right) \geqslant d + 1.$$

Здесь $h^{(k-1)} \in F^{(k-1)}$, *и* $F^{(k-1)} = \left\{f_0^{(k-1)}, f_1^{(k-1)}\right\}$.

Также доказано очевидное следствие этой теоремы.

Следствие 1. *Если* h *— одна из функций, входящая в рассматриваемую последовательность на произвольном шаге, то*

$$\mathrm{AI}(h) \geqslant \Omega(\sqrt{n}),$$

где n *— это количество входов либо степень функции* h.

То есть последовательность степеней обнуляющих функций возрастает с ростом числа входов и составляет $\Omega(\sqrt{n})$, где n — число входов данной булевой функции, либо ее степень. Как уже было сказано, нижних оценок алгебраической иммунности конкретной последовательности функций в печати не встречалось, тем более — растущих нижних оценок.

Список литературы

[1] *Тарасников Ю. В.* О корреляционно-иммунных и устойчивых булевых функциях. Математические вопросы кибернетики, вып. 11, М.: Физматлит, 2002, с. 91–148.

[2] *Тарасников Ю. В.* О новых конструкциях нелинейных фильтров для поточных шифраторов и их устойчивости против стандартных и новых криптографических атак. Математика и безопасность информационных технологий. Материалы конференции в МГУ 23–24 октября 2003 г. М.: МЦНМО, 2004, с. 160–164.

[3] *Meier W.*, *Pasalic E.*, *Carlet C.* Algebraic attack and decomposition of Boolean functions. Proceedings of Eurocrypt 2004, Interlaken, Switzerland, May 2–6, 2004, Lecture Notes in Computer Science, V. 3027, pp. 474–491, Springer-Verlag, 2004.

[4] *Armknecht F.* On the existence of low-degree equations for algebraic attacks. Cryptology ePrint archive (`http://eprint.iacr.org/`), Report 2004/185, August 2004, 16 pp.

[5] *Courtois N.* Higher order correlation attacks, XL algorithm, and cryptanalysis of Toyocrypt. Proceedings of 5th International Conference on Information Security and Cryptology (ICISC 2002), November 28–29, 2002, Seoul, Korea, Lecture Notes in Computer Science, V. 2587, pp. 182–199, Springer-Verlag, 2002.

[6] *Courtois N.*, *Meier W.* Algebraic attacks on stream ciphers with linear feedback. Advances in Cryptology: Eurocrypt 2003, Warsaw, Poland, May 4–8, 2003, Proceedings, Lecture Notes in Computer Science, V. 2656, pp. 345–357, Springer-Verlag, 2003.

[7] *Fedorova M.*, *Tarannikov Yu.* On the constructing of highly nonlinear resilient Boolean functions by means of special matrices. Progress in Cryptology — Indocrypt 2001, Chennai, India, December 16–20, 2001, Proceedings, Lecture Notes in Computer Science, V. 2247, pp. 254–266, Springer-Verlag, 2001.

[8] *Tarannikov Yu.* On resilient Boolean functions with maximal possible nonlinearity. Proceedings of Indocrypt 2000, Calcutta, India, December 10–13, 2000, Lecture Notes in Computer Science, V. 1977, pp. 19–30, Springer-Verlag, 2000.

[9] *Tarannikov Yu.*, New constructions of resilient Boolean functions with maximal nonlinearity. Fast Software Encryption, 8th International Workshop (FSE 2001), Yokohama, Japan, April 2–4, 2001. Revised Papers, Lecture Notes in Computer Science, V. 2355, 2002, pp. 66–77.

[10] *Tarannikov Yu.*, *Korolev P.*, *Botev A.* Autocorrelation coefficients and correlation immunity of Boolean functions. Proceedings of Asiacrypt 2001, Gold Coast, Australia, December 9–13, 2001, Lecture Notes in Computer Science, V. 2248, pp. 460–479, Springer-Verlag, 2001.

О некоторых свойствах уровня аффинности комбинирующих булевых функций

М. Л. Буряков

Одним из основных методов оценки стойкости комбинирующих генераторов является корреляционный метод (см. [5]). Введенная на его основе концепция корреляционной иммунности булевых функций представляет собой свойство, позволяющее комбинирующему генератору противостоять корреляционному методу анализа.

Вместе с тем очевидно, что корреляционно-иммунная функция может иметь некоторые другие криптографические слабости. Одна из таких слабостей описывается параметром — уровнем аффинности булевой функции, введенном в [2].

Введем основные необходимые для дальнейшего изложения обозначения и определения.

Пусть $\mathbb{F}_2$ — поле Галуа из двух элементов. Для произвольного натурального числа $n \in \mathbb{N}$ будем рассматривать векторное пространство $V_n = \mathbb{F}_2^n$. Векторы из V_n будем обозначать жирным шрифтом: $\mathbf{x}$, $\mathbf{y}$ — при этом $x^{(i)}$ — i-я компонента вектора $\mathbf{x}$; $\langle \mathbf{x}, \mathbf{y} \rangle$ — скалярное произведение векторов $\mathbf{x}$ и $\mathbf{y}$.

Множество булевых функций, существенно зависящих от всех n переменных, будем обозначать $\mathcal{F}_n$.

Для произвольного множества переменных $X = \{x^{(1)}, \ldots, x^{(n)}\}$ будем считать, что $f(X)$ обозначает функцию $f\left(x^{(1)}, \ldots, x^{(n)}\right) \in \mathcal{F}_n$, причем переменные следуют в порядке возрастания индексов. Также иногда будем обозначать переменные, от которых зависит функция, как вектор: $f(\mathbf{x})$.

Величина $\deg f$ — алгебраическая степень полинома Жегалкина функции f. Аффинную функцию $f(\mathbf{x}) = \langle \mathbf{x}, \mathbf{a} \rangle \oplus b$ обозначим как $l_{\mathbf{a},\mathbf{b}}$.

Через $f_{i_1, \ldots, i_s}^{\sigma^{(1)}, \ldots, \sigma^{(s)}}$ будем обозначать подфункцию булевой функции $f(x^{(1)}, \ldots, x^{(n)})$ $(s \leqslant n)$, полученную подстановкой констант $\sigma^{(1)}, \ldots, \sigma^{(s)}$ на место переменных $x^{(i_1)}, \ldots, x^{(i_s)}$ соответственно.

Определения корреляционной-иммунности порядка m и m-устойчивости булевых функции см., например, [1].

Определение 1 ([2]). Булева функция $f \in \mathcal{F}_n$ называется *k-аффинной*, $0 \leqslant k \leqslant n-1$, если существуют наборы $1 \leqslant i_1 < \ldots < i_k \leqslant n$, $\mathbf{b} \in V_k$ такие, что $f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}$ является аффинной.

Определение 2 ([2]). *Уровнем аффинности* la f булевой функции $f \in \mathcal{F}_n$ называется минимальное неотрицательное целое число k, для которого f является k-аффинной.

Приведем полученные оценки уровней аффинности для некоторых классов булевых функций. Докажем одно вспомогательное утверждение.

Лемма 1. *Пусть $\pi\colon V_n \to V_n$ — произвольная подстановка на пространстве V_n, представимая покоординатно в виде $\pi(X) = (f_1(X), \ldots, f_n(X))$, где $f_1, \ldots, f_n \in \mathcal{F}_n$. Тогда при любой фиксации любых k переменных $(0 < k \leqslant n)$ в константу обращается не более чем k функций $f_1, \ldots, f_n$.*

Доказательство. Предположим противное, то есть существование таких наборов $1 \leqslant s_1 < \ldots < s_k \leqslant n$ и $\mathbf{b} \in V_k$, что для некоторого $m > k$ выполнено

$$f_{i_1 s_1, \ldots, s_k}^{b^{(1)}, \ldots, b^{(k)}} = c_1, \ldots, f_{i_m s_1, \ldots, s_k}^{b^{(1)}, \ldots, b^{(k)}} = c_m, \quad c_i \in \{0, 1\} \quad (i = \overline{1, m}).$$

Рассмотрим подпространства $L = \{u \in V_n : u^{(s_1)} = \ldots = u^{(s_k)} = 0\}$ и $L' = \{v \in V_n : v^{(i_1)} = \ldots = v^{(i_m)} = 0\}$ пространства V_n.

Имеет место неравенство $\dim L = n - k > \dim L' = n - m$, при этом смежный класс $L \oplus \mathbf{b}$ переводится отображением π в смежный класс $L' \oplus \mathbf{c}$, где $\mathbf{c} = (c_1, \ldots, c_m)$. Мощности этих смежных классов различны, что противоречит биективности отображения π. □

Теорема 1. *Для любой максимально нелинейной функции f от $2n$ переменных из $\mathcal{M}$-класса* la $f = n$.

Доказательство. Максимально нелинейные функции из $\mathcal{M}$-класса (см., например, [1]) представимы в виде

$$f(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \pi(\mathbf{y}) \rangle \oplus \psi(\mathbf{y}), \qquad \mathbf{x}, \mathbf{y} \in V_n,$$

где $\pi\colon V_n \to V_n$ — произвольная подстановка на пространстве V_n и $\psi \in \mathcal{F}_n$ — произвольная булева функция от n переменных.

Рассмотрим функцию

$$f'(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x}, \pi(\mathbf{y}) \rangle = \bigoplus_{i=1}^{n} x^{(i)} f_i(\mathbf{y}),$$

где $f_i \in \mathcal{F}_n$ — некоторые булевы функции $(i = \overline{1, n})$.

Из взаимной однозначности отображения $\pi(X)$ следует, что $\deg f_i \geqslant 1$ для всех $i = \overline{1, n}$, а по Лемме 1 при фиксации l позиций вектора $\mathbf{y}$ в константу обратится не более чем l функций f_i. Отсюда получаем, что la $f' = n$.

Теперь рассмотрим добавление к $f'(\mathbf{x}, \mathbf{y})$ функции $\psi(\mathbf{y})$. При произвольной фиксации вектора $\mathbf{y}$ (n переменных) для функции f мы полу-

чаем линейную функцию, то есть уровень аффинности не увеличится. Уменьшиться он также не может, так как множества переменных векторов **x** и **y** не пересекаются, и для линеаризации всей функции f необходима линеаризация по крайней функции f'. Таким образом, $\mathrm{la}\, f = n$. □

Конструкция Майорана-МакФарланда, рассмотренная в предыдущей теореме, используется также для построения устойчивых функций. Будем считать, что функция f принадлежит классу $\mathcal{M}$. Воспользовавшись координатным представлением $\varphi = (f_1, \ldots, f_r)$, запишем функцию в виде

$$f(X) = f_{\varphi,\psi}(X) = \bigoplus_{i=1}^{r} x^{(i)} f_i\big(x^{(r+1)}, \ldots, x^{(n)}\big) \oplus \psi\big(x^{(r+1)}, \ldots, x^{(n)}\big). \quad (1)$$

Лемма 2 ([4]). *Пусть $f \in \mathcal{F}_n$ вида* (1), *p, r — натуральные числа такие, что $n \geqslant r > p \geqslant 0$, отображение $\varphi\colon V_{n-r} \to V_r$ удовлетворяет условию $\mathrm{wt}(\varphi(u)) \geqslant p$ для любого $u \in V_{n-r}$, ψ — произвольная функция из $\mathcal{F}_{n-r}$. Тогда f — m-устойчива для некоторого $m \geqslant p$.*

Предложение 1. *Для булевой функции f, удовлетворяющей условиям Леммы* 2, *выполняется неравенство* $\mathrm{la}\, f \leqslant n - r$.

Доказательство. Справедливость утверждения непосредственно следует из вида функции. □

Необходимо отметить, что при $n - r < p \leqslant m$ для функций данного класса, использующихся в комбинирующем генераторе, мы получаем возможность опробовать на первом этапе метода анализа, использующего ранговый критерий, начальные состояния меньшего числа регистров нежели при реализации корреляционного метода.

Теорема 2. *Для m-устойчивой булевой функции f, достигающей верхней границы нелинейности, построенной по конструкции, предложенной в* [3, § 6, *Теорема* 7],

$$\mathrm{la}\, f \leqslant \left\lfloor \frac{m+3}{2} \right\rfloor.$$

Доказательство. При доказательстве будем использовать обозначения, принятые в [3].

Функция f строится как

$$f(x_1, \ldots, x_n) = f_{3(n-m)-7,1}(x_1, \ldots, x_{3(n-m)-7}) \bigoplus_{i=3(n-m)-6}^{n} x_i,$$

где $f_{3(n-m)-7,1}$ — булева функция от $3(n-m)-7$ специального вида и строящаяся рекурсивно (см. [3, § 6, Леммы 28, 29, Следствие 7]). При этом $n = 3k - 7$, $m = 2k - 7$, где $k \geqslant 3$ — шаг рекурсии.

Очевидно, что $\mathrm{la}\, f = \mathrm{la}\, f_{3(n-m)-7,1}$.

Рассмотрим конструкцию функции $f_{3(n-m)-7,1}$. Для ее построения используется шаг рекурсивного перехода от n к $n+2$. Здесь вводятся функции

$$f_1(x_1, \ldots, x_{n+2}) = f_{n,1}(x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_{j-1}, x_{j+1}, \ldots, x_{n+2}) \oplus \oplus x_i \oplus x_j,$$

$$f_2(x_1, \ldots, x_{n+2}) = f_{n+1,2}(x_1, \ldots, x_{n+1}) \oplus x_{n+2},$$

из которых получаем

$$f_{n+3,1} = (x_{n+3} \oplus 1) f_1(x_1, \ldots, x_{n+2}) \oplus f_2(x_1, \ldots, x_{n+2}),$$

$$f_{n+4,2} = (x_{n+3} \oplus x_{n+4} \oplus 1) f_1(x_1, \ldots, x_{n+1}) \oplus \oplus (x_{n+3} \oplus x_{n+4}) f_2(x_1, \ldots, x_{n+2}) \oplus x_{n+3}.$$

Нетрудно показать, что при этом

$$\mathrm{la}\, f_{n+3,1} \leqslant \min\{\mathrm{la}\, f_1, \mathrm{la}\, f_2\} + 1 = \min\{\mathrm{la}\, f_{n,1}, \mathrm{la}\, f_{n,2}\} + 1.$$

То есть на каждом шаге рекурсии уровень аффинности может увеличиться не более чем на 1.

При $k = 3$ имеем $\mathrm{la}\, f_{2,1} \leqslant 1$. Отсюда следует, что $\mathrm{la}\, f \leqslant k - 2$ для всех $k \geqslant 3$, и учитывая, что $m = 2k - 7$, получаем, что $\mathrm{la}\, f \leqslant \left\lfloor \frac{m+3}{2} \right\rfloor$. □

Заметим, что при выборе в базисе рекурсии функций, использованных в работе [3], уровень аффинности получаемой m-устойчивой функции в точности равен $\left\lfloor \frac{m+3}{2} \right\rfloor$.

Теорема 2 показывает, что при достаточно жестких ограничениях на функцию, обусловленных стремлением противостоять корреляционному методу анализа, уровень аффинности этих функций может оказываться достаточно низким (в данном случае он по порядку в два раза меньше уровня корреляционной иммунности).

Практически полностью аналогично Теореме 2 можно доказать справедливость следующего утверждения.

Теорема 3. *Для функций $f^k_{n,1}$ и $f^k_{n,2}$, предложенных в* [3, §7, *Лемма* 30], *уровень аффинности не превосходит величины* $2(k-2)$.

Список литературы

[1] *Логачёв О. А., Сальников А. А., Ященко В. В.* Булевы функции в теории кодирования и криптологии. — М.: МЦНМО, 2004.

[2] *Логачёв О. А., Сальников А. А., Ященко В. В.* Комбинирующие k-аффинные функции. Математика и безопасность информационных техно-

логий. Материалы конференции в МГУ 23–24 октября 2003 г. — М.: МЦНМО, 2004, сс. 176–178.

[3] *Тарасников Ю. В.* О корреляционно-иммунных и устойчивых булевых функциях. Математические вопросы кибернетики. Вып. 11, — М.: Физматлит, 2002, сс. 91–148.

[4] *Carlet C.* A Large Class of Cryptographic Boolean Functions via a Study of the Maiorana-McFarland Constructions. Advances in Cryptology: CRYPTO'02. Lect. Notes in Comput. Sci. V. 2442. — New York: Springer-Verlag, 2002, pp. 549–564.

[5] *Siegenthaler T.* Correlation-immunity of Nonlinear Combining Functions for Cryptographic Applications. IEEE Trans. on Information Theory IT-30(5): 776–780, 1984.

О распределении уровня аффинности на множестве булевых функций

М. Л. Буряков, О. А. Логачёв

Параметр — уровень аффинности булевой функции был введен в [1]. В работах [1], [2] был рассмотрен ряд свойств этого параметра и его связи с другими криптографическими свойствами булевых функций. В настоящей работе предложены эти исследования.

Будем обозначать $\mathbb{F}_2$ — конечное поле из двух элементов, $V_n = \mathbb{F}_2^n$ — пространство вектор-столбцов над полем $\mathbb{F}_2$, $\mathcal{F}_n$ — множество булевых функций, то есть множество всех возможных отображений из V_n в $\mathbb{F}_2$. Через $\mathcal{F}_n^*$ будем обозначать множество булевых функций из $\mathcal{F}_n$, существенно зависящих от всех n переменных. Через $\deg(f)$ будем обозначать степень алгебраической нормальной формы (а. н. ф.) функции f из $\mathcal{F}_n$. Для $\mathbf{x} \in V_n$ и $f \in \mathcal{F}_n$ через $\mathrm{wt}(x)$ и $\mathrm{wt}(f)$ соответственно обозначим их веса. Тогда $\mathrm{dist}(x, y) = \mathrm{wt}(\mathbf{x} \oplus \mathbf{y})$ для любых $\mathbf{x}, \mathbf{y} \in V_n$, и $\mathrm{dist}(f_1, f_2) = \mathrm{wt}(f_1 \oplus f_2)$ для любых $f_1, f_2 \in \mathcal{F}_n$ ($\oplus$ — покомпонентное сложение векторов по mod2). Если $\deg(f) \leqslant 1$, то функция f называется аффинной. Множество таких функций в $\mathcal{F}_n$ обозначим через $\mathcal{A}_n$. Пусть $f \in \mathcal{F}_n$; для наборов

$$1 \leqslant i_1 < \ldots < i_k \leqslant n, \quad \mathbf{b} = \left(b^{(1)}, \ldots, b^{(k)}\right)^T \in V_k,$$

неотрицательного $k \leqslant n$ обозначим через $f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}$ булеву функцию из $\mathcal{F}_{n-k}$, полученную из f фиксацией переменных $x^{(i_1)} = b^{(1)}, \ldots\ x^{(i_k)} = b^{(k)}$.

Определение 1. Булева функция f из $\mathcal{F}_n^*$ называется k-аффинной, $0 \leqslant k \leqslant n-1$, если существуют наборы

$$1 \leqslant i_1 < \ldots < i_k \leqslant n, \quad \mathbf{b} = \left(b^{(1)}, \ldots,\ b^{(k)}\right)^T \in V_k$$

такие, что $f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}$ является аффинной функцией, то есть

$$\deg\left(f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}\right) \leqslant 1.$$

Таким образом, ясно, что 0-аффинными функциями в $\mathcal{F}_n^*$ будут две функции: $x^{(1)} \oplus x^{(2)} \oplus \ldots x^{(n)}$ и $x^{(1)} \oplus x^{(2)} \oplus \ldots x^{(n)} \oplus 1$.

Работа второго автора поддержана Российским фондом фундаментальных исследований (номера проектов 02-01-00581 и 02-01-00687).

Определение 2. Уровнем аффинности $\mathrm{la}\ f$ булевой функции $f \in \mathcal{F}_n^*$ называется минимальное неотрицательное число k, для которого f является k-аффинной.

Для любой функции f из $\mathcal{F}_n^*$ справедливо неравенство $\mathrm{la}\ f \leqslant n-1$.

Параметр $\mathrm{la}\ f$ не является инвариантом относительно действия на функцию f элементов полной аффинной группы $\mathfrak{GA}(V_n)$.

Пример 1. Пусть

$$f(x) = f\big(x^{(1)}, x^{(2)}, x^{(3)}, x^{(4)}\big) = x^{(1)} \cdot \big(x^{(2)} \oplus x^{(3)} \oplus x^{(4)}\big).$$

Ясно, что $\mathrm{la}\ f = 1$. Рассмотрим элемент $\mathfrak{g}$ группы $\mathfrak{GA}(V_n)$ вида $\mathfrak{g} = (A, c)$, где

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad c = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

Тогда $f'(x) = f^{\mathfrak{g}}(x) = f(Ax \oplus c) = \big(x^{(1)} \oplus x^{(4)}\big) \cdot \big(x^{(2)} \oplus x^{(3)} \oplus x^{(4)}\big)$. Легко вычислить, что $\mathrm{la}\ f' = \mathrm{la}\ f^{\mathfrak{g}} = 2$.

Интересна связь, которая имеется у данного криптографического параметра с другим криптографическим параметром, также не являющимся инвариантом относительно группы $\mathfrak{GA}(V_n)$.

Определение 3. Булева функция $f \in \mathcal{F}_n$ называется корреляционно-иммунной порядка r, $1 \leqslant r \leqslant n$, если для любых наборов

$$1 \leqslant j_1 < \ldots < j_r \leqslant n, \quad \mathbf{a} = \big(a^{(1)}, \ldots, a^{(r)}\big)^T \in V_r$$

выполняется соотношение

$$\mathrm{wt}\big(f_{j_1, \ldots, j_r}^{a^{(1)}, \ldots, a^{(r)}}\big) = \frac{\mathrm{wt}(f)}{2^r},$$

при этом функция $f_{j_1, \ldots, j_r}^{a^{(1)}, \ldots, a^{(r)}}$ рассматривается как функция из $\mathcal{F}_{n-r}$.

С учетом приведенного определения естественно вводится (см. [3]) следующее обозначение

$$\mathrm{cor}\ f = \max\{r \in \mathbb{N} : f \text{ — корреляционно-иммунна порядка } r\},$$

где $\mathbb{N}$ — множество натуральных чисел.

Необходимо отметить, что в определении 3 не требуется, чтобы функция f принадлежала множеству $\mathcal{F}_n^*$. Однако везде ниже мы будем рассматривать именно такие функции.

Рассмотрим следующее утверждение.

Теорема 1. *Пусть* $f \in \mathcal{F}_n^*$, $\mathrm{cor}\ f > 0$, *и* $\mathrm{wt}(f) \neq 2^i$, $i = 0, 1, \ldots, n-1$. *Тогда* $\mathrm{la}\ f > \mathrm{cor}\ f$.

Доказательство. Предположим противное, то есть существование $k > 0$ такого, что

$$k = \mathrm{la}\ f \leqslant \mathrm{cor}\ f = r.$$

Из условия теоремы следует, что вес функции f можно представить в виде $\mathrm{wt}(f) = (2t+1)2^s$ для некоторых подходящих неотрицательных t, $t \geqslant 1$ и s. Согласно предположению существуют наборы

$$1 \leqslant i_1 < \ldots < i_k \leqslant n, \quad \mathbf{b} = \big(b^{(1)}, \ldots, b^{(k)}\big)^T \in V_k$$

такие, что $f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}} \in \mathcal{A}_{n-k}$. Кроме того, поскольку $k \leqslant r$, то функция f является корреляционно-иммунной порядка k и

$$\mathrm{wt}\big(f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}\big) = \frac{\mathrm{wt}(f)}{2^k} = \frac{(2t+1)2^s}{2^k},$$

Следовательно, $\mathrm{wt}\big(f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}\big)$ делится на нечетное число $2t+1$. С другой стороны, поскольку $\mathrm{wt}\big(f_{i_1, \ldots, i_k}^{b^{(1)}, \ldots, b^{(k)}}\big)$ аффинная функция из $\mathcal{A}_{n-k}$ ее вес можно принимать значения 0, 2^{n-k-1}, 2^{n-k}. Полученное противоречие доказывает утверждение теоремы. □

Пусть $\mathcal{U}_n$ — множество уравновешенных булевых функций из $\mathcal{F}_n$, $\#\,\mathcal{U}_n = \binom{2^n}{2^{n-1}}$. Обозначим через $\mathcal{Q}_n = \mathcal{U}_n \cap \mathcal{F}_n^*$ множество функций из $\mathcal{U}_n$, существенно зависящих от всех переменных. Воспользовавшись методом включения-исключения легко получить следующее равенство:

$$q_n = \#\mathcal{Q}_n = \sum_{s=0}^{n-1} (-1)^s \binom{n}{s} \binom{2^{n-s}}{2^{n-s-1}}. \tag{1}$$

Ниже нам потребуется достаточно грубая оценка для q_n. Рассмотрим для каждого $j = 1, 2, \ldots, n$ подмножество $\mathcal{M}_j$ множества $\mathcal{Q}_n$, состоящее из функций вида

$$g(x^{(1)}, \ldots, x^{(n)}) = g'(x^{(1)}, \ldots, x^{(j-1)}, x^{(j+1)}, \ldots, x^{(n)}) \oplus x^{(j)},$$

где $\deg(g') = n-1$ и в алгебраической нормальной форме g' нет мономов степени 1. Так как $\mathcal{M}_{j_1} \cap \mathcal{M}_{j_2} = \varnothing$, $j_1 \neq j_2$ и $\#M_j = 2^{2^{n-1}-n}$ для любого $j = 1, 2, \ldots, n$, то

$$q_n \geqslant n 2^{2^{n-1}-n}. \tag{2}$$

Множество Q_n разбивается на непересекающиеся подмножества

$$Q_{ni} = \{f \in \mathcal{Q}_n : \ \mathrm{la}\ f = i\},$$

$i = 0, 1, \ldots, n-1$, $\#\mathcal{Q}_{ni} = q_{ni}$. Следовательно

$$Q_n = Q_{n0} \cup Q_{n1} \cup \ldots \cup Q_{n,n-1} \tag{3}$$

и

$$q_n = q_{n0} + q_{n1} + \ldots + q_{n,n-1}. \tag{4}$$

Множество Q_{n0}, как указывалось ранее, содержит две функции и $q_{n0}=2$. Рассмотрим теперь верхнюю оценку для $q_{n,n-1}$. Для этого нам потребуется следующее утверждение.

Теорема 2. *Для булевой функции f из $\mathcal{F}_n^*$ соотношение* $\operatorname{la} f = n-1$ *выполнено тогда и только тогда, когда*

$$f\left(x^{(1)}, \ldots, x^{(n)}\right) = \bigoplus_{i<j} x^{(i)}x^{(j)} \oplus l_{ab}\left(x^{(1)}, \ldots, x^{(n)}\right), \qquad (5)$$

где $l_{ab}\left(x^{(1)}, \ldots, x^{(n)}\right) = \bigoplus_{i=1}^{n} a^{(i)}x^{(i)} \oplus b$, $\mathbf{a} \in V_n$, $b \in \mathbb{F}_2$.

Доказательство. Необходимость. Пусть $f \in \mathcal{F}_n^*$ и $\operatorname{la} f = n-1$. Для произвольной перестановки $(i_1, i_2, \ldots, i_n)$ элементов множества $\{1, 2, \ldots, n\}$ рассмотрим фиксацию переменных $x^{(i_1)}=0, \ldots, x^{(i_{n-2})}=0$. Данная фиксация превращает в нуль любой моном, степень которого не менее 2, за исключением монома $x^{(i_{n-1})}x^{(i_n)}$, остающегося неизменным. Следовательно, в а. н. ф. функции f присутствует моном $x^{(i_{n-1})}x^{(i_n)}$. Поскольку рассуждения справедливы для любой перестановки $(i_1, i_2, \ldots, i_n)$, то в а. н. ф. функции присутствуют все $n(n-1)/2$ мономов степени 2.

Предположим теперь, что в а. н. ф. функции f присутствует моном вида $x^{(j_1)}x^{(j_2)}x^{(j_3)}$, $j_1 < j_2 < j_3$. Рассмотрим следующую фиксацию $n-2$ переменных. Положим $x^{(i)}=0$, $i \neq j_1, j_2, j_3$, $x^{(j_3)}=1$. При подстановке этой фиксации переменных в f превращаются в нуль все мономы степени 3 и выше, за исключением монома $x^{j_1}x^{j_2}x^{j_3}$, который переходит в $x^{j_1}x^{j_2}$. Мономы второй степени при такой подстановке превращаются в мономы степени не превосходящей 1, за исключением монома $x^{(j_1)}x^{(j_2)}$, который остается неизменным. Таким образом мы получаем функцию вида $x^{(j_1)}x^{(j_2)} \oplus x^{(j_1)}x^{(j_2)} \oplus a^{(j_1)}x^{(j_1)} \oplus a^{(j_2)}x^{(j_2)} \oplus b$, $a^{(j_1)}$, $a^{(j_2)}$, $b \in \mathbb{F}_2$. Таким образом $\operatorname{la} f = n-2$, что противоречит нашему предположению. Следовательно, монома $x^{(j_1)}x^{(j_2)}x^{(j_3)}$ нет в а. н. ф. функции f. Поскольку тройка (j_1, j_2, j_3) выбирается произвольно, то нами показано, что в а. н. ф. функции f нет мономов степени 3. Аналогичными рассуждениями можно показать, что в а. н. ф. функции f нет мономов и степеней больших 3, и функция f имеет вид (5).

Достаточность. Пусть f из $\mathcal{F}_n^*$ имеет вид (5). Из структуры функции легко видеть, что для любых наборов

$$1 \leqslant i_1 < \ldots < i_{n_2} \leqslant n, \quad \mathbf{a} = \left(a^{(1)}, \ldots, a^{(n-2)}\right)^T \in V_{n-2}$$

имеем

$$\deg\left(f_{i_1, \ldots, i_{n-2}}^{a^{(1)}, \ldots, a^{(n-2)}}\right) = 2.$$

Поэтому $\operatorname{la} f > n-2$. Однако, с другой стороны, уровень аффинности любой функции из $\mathcal{F}_n^*$ не превышает $n-1$. Следовательно, $\operatorname{la} f = n-1$. □

Следствие 1. $q_{n,n-1} < 2^{n+1}$.

Доказательство. Это неравенство вытекает из того факта, что не все функции вида (5) являются уравновешенными. □

Следствие 2. $\lim_{n\to\infty} q_{n,n-1}/q_n = 0$.

Доказательство. Равенство вытекает из следствия 1 и неравенства (2). □

Предположения характере поведения величин $\{q_{ni} : i = 0, 1, \ldots, n-1\}$ в настоящее время не могут быть в достаточной мере обоснованы. Однако, представляется верным следующее высказывание.

Гипотеза 1.

a) (слабая форма). Существует натуральное n_0 такое, что при $n \geqslant n_0$

$$(q_{n0} + q_{n1} + \ldots + q_{nt})/q_n \geqslant c,$$

где $t = \left[\dfrac{n+1}{2}\right]$, $0.5 < c \leqslant 1$.

b) (сильная форма). Справедливо соотношение

$$\lim_{n\to\infty} (q_{n0} + q_{n1} + \ldots + q_{nt})/q_n = c,$$

где $t = \left[\dfrac{n+1}{2}\right]$, $0.5 < c \leqslant 1$.

Данная гипотеза в случае доказательства ее справедливости приводит к интересным качественным выводам в области криптологии. Мы попытаемся проиллюстрировать это на примере блокового шифра.

С точки зрения криптоаналитика блоковый шифр — это отображение векторных пространств над полем из двух элементов вида

$$\Phi\colon V_m \times V_n \to V_m$$

такое, что при любой фиксации второго аргумента $\mathbf{k} \in V_n$ отображение $\Phi(\cdot, \mathbf{k})\colon V_m \to V_m$ биективно, то есть реализует некоторую подстановку элементов пространства V_m. При этом если $\Phi(\mathbf{x}, \mathbf{k}) = \mathbf{y}$, то первый аргумент $\mathbf{x} \in V_m$ отображения Φ называют блоком открытого текста (или просто открытым текстом), второй аргумент $\mathbf{k} \in V_n$ называют ключом, а результат $\mathbf{y} \in V_m$ действия отображения Φ называют блоком шифрованного текста (или просто шифрованным текстом). Пусть $(f_1, f_2, \ldots, f_m)$ — координатные функции отображения Φ, то есть

$$y^{(i)} = f_i\left(x^{(1)}, \ldots, x^{(m)}, k^{(1)}, \ldots, k^{(n)}\right), \quad i = 1, 2, \ldots, m.$$

В процессе синтеза блокового шифра Φ используются разнообразные приемы построения раундовых преобразований, широкие классы

отображений (так называемые S-боксы), выбирается индивидуальная (для каждого шифра) схема ключевой развертки. Для построенных таким образом координатных функций шифра Ф легко вычислять значения, но описать их каким-либо другим способом (кроме раундового) не представляется возможным. Все это с достаточно высокой степенью обоснованности позволяет сделать вывод о том, что любая координатная функция, уравновешенная и существенно зависящая от всех $m+n$ переменных, выбирается случайно и равновероятно из Q_{m+n}.

Пусть f — одна из координатных функций шифра Ф. В случае справедливости приведенной выше гипотезы с вероятностью большей 0.5 имеем $\mathrm{la}\ f \leqslant \left[\dfrac{m+n+1}{2}\right]$.

Список литературы

[1] *Логачёв О. А.*, *Сальников А. А.*, *Ященко В. В.* Комбинирующие k-аффинные функции. Математика и безопасность информационных технологий. Материалы конференции в МГУ 23–24 октября 2003 г. — М.: МЦНМО, 2004, сс. 176–178.

[2] *Буряков М. Л.* О некоторых свойствах уровня аффинности комбинирующих булевых функций. Математика и безопасность информационных технологий. МГУ, 28–29 октября 2004 г.

[3] *Логачёв О. А.*, *Сальников А. А.*, *Ященко В. В.* Булевы функции в теории кодирования и криптологии. — М.: МЦНМО, 2004.

Статистическое описание звукового стегоконтейнера

Н. П. Варновский, Е. А. Голубев, О. А. Логачёв

Под естественным стегоконтейнером понимается сформированный амплитудно-цифровым преобразователем (АЦП) файл, содержащий полную информацию об отрезке звукового или визуального образа. Например, это запись в цифре музыкального произведения или речи диктора, полученные с использованием микрофона, радиоприемника или CD-диска. Или последовательность пикселов картинки, полученной цифровой фотокамерой. Файл это зафиксированная реальность, но в цифровом виде. В дальнейшем, с целью экономии ресурса, его можно сжимать с потерями или без потерь, даже синтезировать математическими операциями очень похожие образцы.

Но сначала надо понять, что же такое цифровое отображение реальности, воспринимаемой человеком органами зрения и слуха. Рассмотрим, не ограничивая общности, отображение звука в цифру.

Специалисты по речевым технологиям достигли успехов в анализе и синтезе речи, несмотря на то, что психофизиология речи (и зрения также) недостаточно изучена и понятна для точного моделирования. В основу речевых технологий положено представление звукового процесса в ортогональном базисе гармонических функций. Фурье-преобразования наиболее близки в различных приложениях к среде обитания человека и с успехом используются в звуковых и визуальных технологиях обработки звука и изображения.

Но наша задача состоит не в различении или синтезе звуков или изображений, а в разработке математических, а в дальнейшем, приборных методов оценки естественности файла. Использование файла как контейнера для скрытия информации, т. е. стегоконтейнера, должно проявляться как изменение параметров, описывающих естественность файла.

Поставим задачу разработки математического описания естественного файла, не зависящего от его информационного содержания. Могут

ли быть построены критерии, инвариантные к источнику звуковой или визуальной информации, но определяющие границы флуктуаций параметров этих критериев, при условии, что в файле, содержащем эту информацию, нет ничего постороннего?

Рассмотрим более подробно роль количества разрядов АЦП, преобразующего аналоговый сигнал в последовательность чисел — значений отсчетов при дискретизации и квантовании процесса, протекающего во времени. Для обеспечения разборчивости речи достаточно скорости вокодера 1200 бит/с. При скорости 9,6 Кбит/с обеспечивается узнаваемость собеседника. Для высококачественного воспроизведения музыки требуется скорость более 300 Кбит/с. При представлении одного отсчета звука (или цвета) в виде байта хранение и передача каждого его разряда имеет одну и ту же цену. Но по значимости, как следует из логарифмического правила квантования, разряды имеют разный вес. Младшие разряды отсчетов «заведуют» относительно небольшой частью энергии преобразуемого в цифру сигнала-процесса, но от их содержания зависит качество воспроизведения. Можно ли пренебрегать содержимым младших разрядов как избыточным?

Проведем эксперимент. Обнулим в звуковых файлах все разряды, кроме первого (определяет знак отсчета) и последнего, наименее значимого бита (НЗБ). Так обозначают младший разряд в литературе по компьютерной стеганографии. Затем установим величину разрядов на месте «1» в НЗБ таким образом, чтобы средний уровень величин отсчетов равнялся среднему уровню исходных файлов. Прослушаем, как звучат НЗБ. Иногда — это похожее на шум. Иногда можно различить ритм. Иногда можно различить исполнителя — мужчина или женщина, речь или музыкальное произведение. НЗБ содержат часть информации об информационном содержании файла в целом. НЗБ нельзя считать шумовыми разрядами.

Используем последовательности одноименных разрядов как автономные синхронизированные потоки парциальных энергий оцифрованного сигнала, связанные статистическими зависимостями. Эти зависимости определяют информационное содержание файла. Существуют ли связи между парциальными энергиями, не зависящими от информационного содержания файла, но устойчивые в числовом выражении для всех звуковых файлов, независимо от их источника?

Представим модель файла в виде потоков парциальных энергий, естественным образом формируемых АЦП (рис. 1). Это не ортогональный базис. Это разложение сигнала на его парциальные составляющие, которые в цифроаналоговом преобразователе (ЦАП) полностью восстанавливаются в исходный процесс.

Экспериментальные исследования звукового контейнера

Закроем содержание любого разряда и поставим вопрос: каково содержание (0 или 1) этого разряда? Для этого мы должны построить систему статистического описания, т. е. выбрать критерии, обосновать их выбор, определить объем контролируемого по этим критериям материала. Если эта система соответствует рекомендациям К. Шеннона, то можно быть уверенным в результатах.

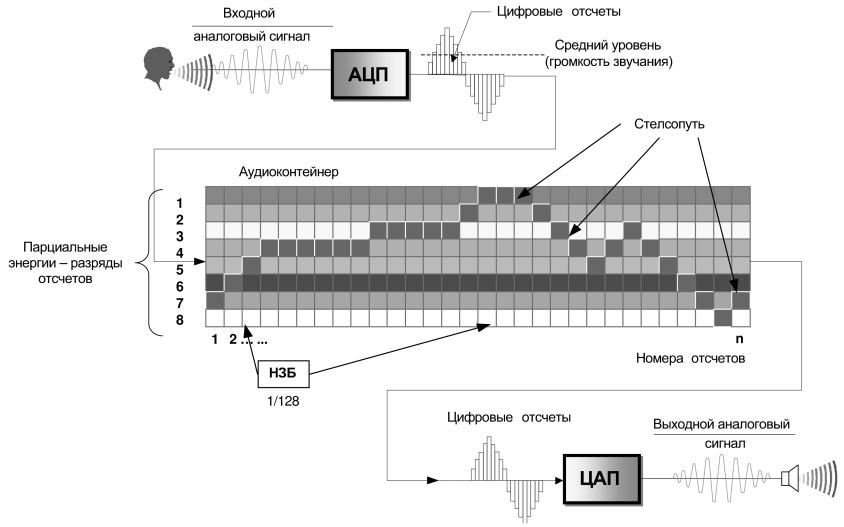


Рис. 1. Модель звукового стегоконтейнера.

Допустим, относительно этой системы получен равновероятный ответ, т. е. нельзя отдать предпочтение 0 или 1. Отметим этот разряд как пригодный для размещения в нем одного бита скрываемой информации. Получим таким образом цепочку разрядов (на рис. 1 обозначены стрелками. Использование любого НЗБ для размещения одного бита сообщения не обнаруживается. Какова допустимая загрузка или вместимость НЗБ, не обнаруживаемая системой статистического описания? Говорить о пропускной способности стеганографического канала связи преждевременно, т. к. файл — стегоконтейнер — пока никуда не передается. С позицией непроявления при прослушивании файла очевидно требование по минимизации изымаемой из файла энергии естественного происхождения и замещении ее на энергию скрываемого сообщения.

При прослушивании скрываемое сообщение звучит как маленькая дудочка в мощном оркестре, но она издает не мелодию, а шум, которого нет в партитуре композитора.

Идея состоит в следующем. Физический аудио- или видеопроцесс разделяется на два процесса: процесс вдоль НЗБ и процесс-файл за вычетом НЗБ. В практической стеганографии — это НЗБ. Иногда в пикселах видео используют НЗБ и более старшие разряды, используется только один цвет и другие приемы, затрагивающие минимальную энергию сигнала.

Разделяя файл-стегоконтейнер на два процесса, мы переходим в область решаемых математической статистикой задач. Статистические зависимости между двумя процессами можно исследовать с применением статистик χ^2, Спирмена, Кендалла, коэффициента корреляции. Эти статистики аддитивные в отношении времени или номеров отсчетов.

Проведем ряд экспериментов на модели звукового стегоконтейнера с целью формирования системы статистических критериев его естественности. В качестве модели будем использовать звуковой процесс, оцифровка которого производится восьмиразрядным АЦП с частотой

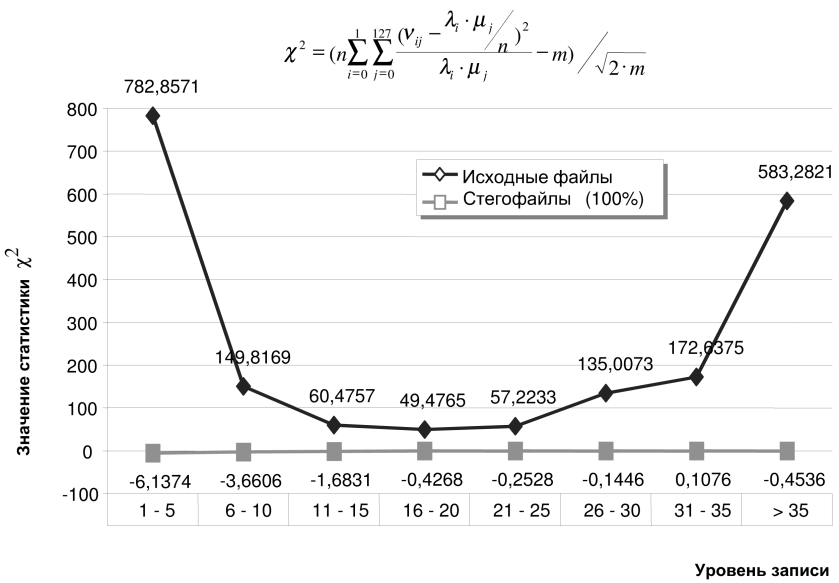


Рис. 2. Сравнение средних значений статистики χ^2 для CD-записей до и после скрытия.

дискретизации 44100 раз в секунду. При этих параметрах продолжительность звучания должна превышать 3 секунды для корректного применения статистических методов. В эксперименте продолжительность звучания выбрана в 15 секунд, т. е. более 600 тысяч отсчетов.

Выделим в модели (см. рис. 1) в качестве мест размещения защищаемой информации наименее значимые биты. Вычислим статистику χ^2, для исходного файла (рис. 2). В областях слабого и очень сильного сигнала статистические связи усиливаются. Но эти области практически не используются при записи звука. В областях тихого, среднего и громкого звучания функция χ^2 имеет отличающиеся от нуля значения. При полной загрузке НЗБ (100%) значение χ^2 близко к нулю для любого уровня сигнала. Аналогичный результат на том же материале получается при вычислении коэффициента корреляции двух процессов (рис. 3).

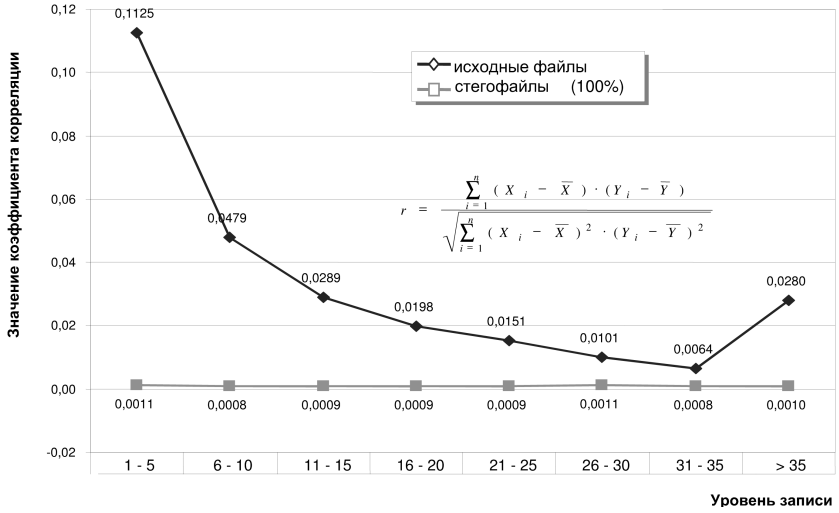


Рис. 3. Сравнение средних значений коэффициента корреляции для CD-записей до и после скрытия.

Таким образом, выбранные статистические критерии позволяют однозначно различать чистые и загруженные контейнеры.

Экспериментально установлено, что для любых источников и рабочих значений громкости распределение значений функции χ^2 описывается нормальным распределением с параметрами m и $\sigma = (1/3)m$, при этом m слабо зависит от конкретного источника. Изучено влияние частичной загрузки НЗБ звукового файла на поведение функции χ^2 (рис. 3).

Допустим, по какому правилу устанавливается порог принятия решения. Эти правила могут соответствовать минимаксному критерию, Неймана — Пирсона и т. п. Ясно одно — при уменьшении загрузки стелсопути решение о наличии скрываемой информации будет статистическим решением:

- $P_{\text{л. т.}}$ — вероятность ложного подозрения (тревоги),
- $P_{\text{пропуска}}$ — вероятность необнаружения стеганографического заражения информации,
- $P_{\text{прав.}}$ — правильное решение о чистоте или зараженности.

Выбор правила принятия решения и практические меры, следующие из этого решения, определяются системными требованиями к стеганографической системе.

У функции χ^2 есть одно свойство — она аддитивна и не описывает развитие процесса во времени. Если переставить местами отсчеты в естественном звуковом файле, то значение χ^2 не изменится.

Необходимо описание звукового процесса, развивающегося во времени. Интерес представляет изучение статистических характеристик НЗБ, т. к. в этом разряде происходит нарушение естественности. Здесь

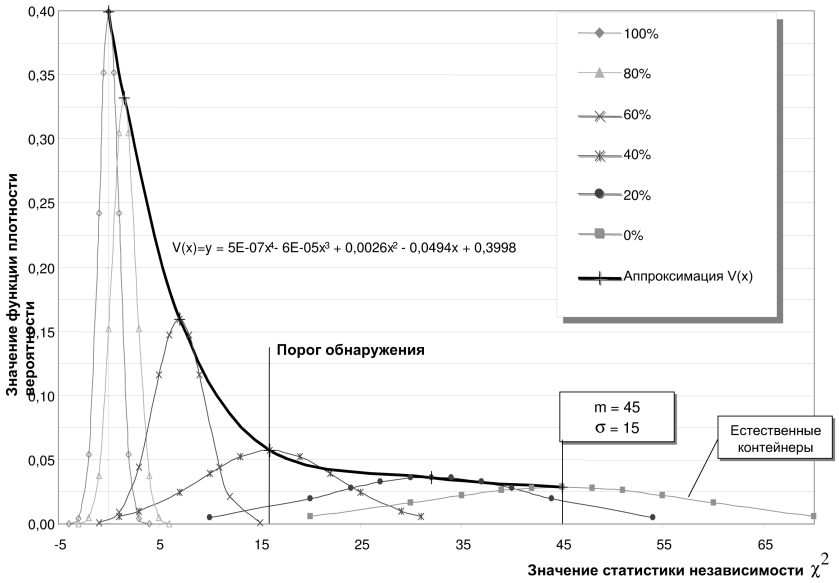


Рис. 4. Поведение функции χ^2 в зависимости от загрузки аудиоконтейнера.

мы также находимся в рамках задач математической статистики, в частности Марковских процессов.

Выберем два параметра: распределение переходов состояний разрядов соседних отсчетов в порядке их возрастания — это частости переходов $1 \to 1$, $1 \to 0$, $0 \to 1$, $0 \to 0$; и распределение длин серий 0 и 1.

Исследование частости переходов показало их качественную статистическую устойчивость, иллюстрируемую на рис. 4.

Старшие разряды имеют большую по времени память, которая убывает с номером разряда. В НЗБ всегда преобладают переходы $1 \to 1$ и $0 \to 0$, при размещении в НЗБ псевдослучайной последовательности гистограмма выравнивается. Это является еще одним признаком наличия стелсозаражения НЗБ.

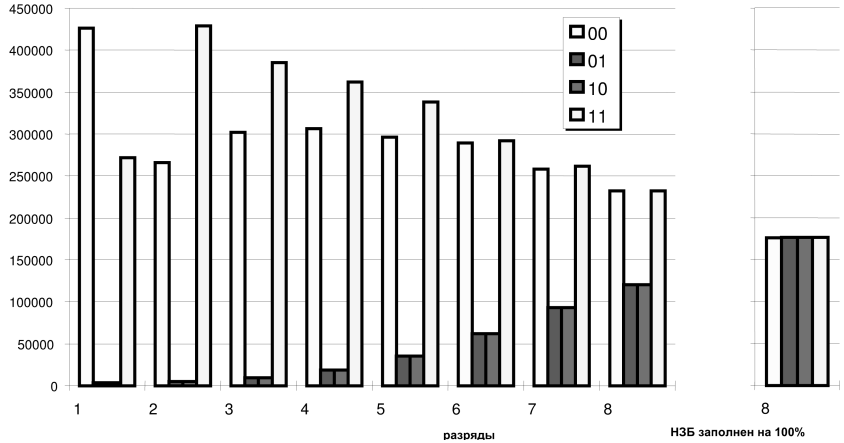


Рис. 5. Частота переходов $0 \to 0$, $0 \to 1$, $1 \to 0$, $1 \to 1$ в разрядах естественного аудиофайла; $n \approx 600\,000$, продолжительность звучания 15 с.

В криптографии существуют алгоритмы, которые за счет небольшой избыточности придают ПСП свойства марковости. Применяя их, можно добиться распределения переходов в восьмых разрядах, похожих на распределение переходов в естественном процессе. Но критерий по χ^2 для файла в целом выявляет применение этого способа.

Параметр, описывающий разность между количеством переходов $1 \to 1$, $0 \to 0$ и количеством переходов $0 \to 1$, $1 \to 0$ для звуковых файлов имеет также нормальное распределение.

Типовое распределение длин серий в НЗБ приведено на рис. 6.

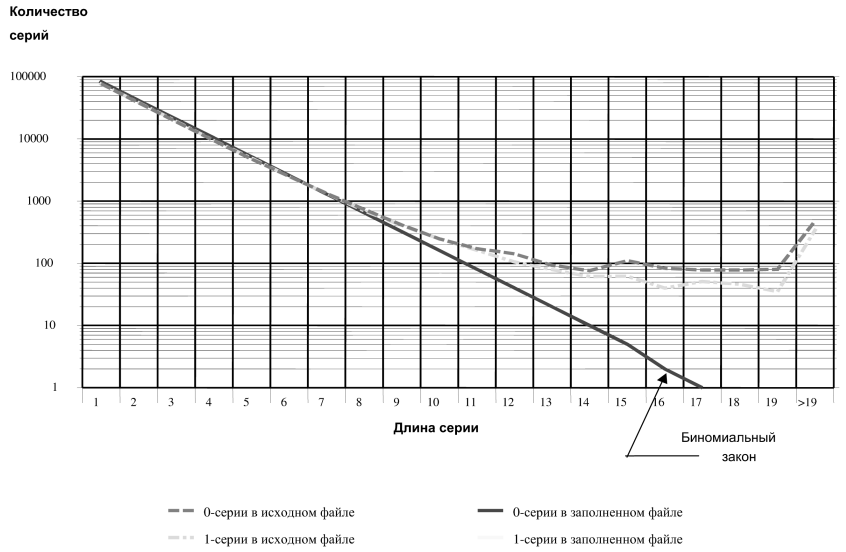


Рис. 6. Распределение длин серий естественного аудиоконтейнера.

Появление длин серий нулей и единиц, отличных от биномиального закона, описывающего частость длин серий случайной последовательности, полученной от идеального генератора случайных чисел, является признаком естественности потока НЗБ. Хорошая система адресации не должна разрушать параметры этого признака.

В результате исследований можно сформулировать следующие выводы.

1. Одним из конструктивных направлений математического описания, звукового или визуального образа, представленного в цифровом виде в форме файла, является интерпретация этого файла как синхронизированных потоков парциальных энергий, зафиксированных в одноименных разрядах цифровых отсчетов. Файл — это физический процесс во времени.

2. Стеганографическое скрытие информации интерпретируется как изъятие части энергии естественного процесса и замещение ее энергией замещающего процесса. Файл как бы расщепляется на два процесса: НЗБ — процесс, в котором может скрываться информация, и файл за вычетом НЗБ, который используется для прикрытия, маскировки первого процесса.

3. Разделение файла на два физических процесса вводит проблему в предметную область математической статистики и других математических дисциплин, описывающих и исследующих физические процессы.

4. Проведен первый этап формирования базовой модели естественности и нарушения естественности аудио- или видеофайла, реализующий, но не в полной мере, рекомендации К. Шеннона. Выбраны три критерия, реагирующие на стеганографическую маскировку информации и инвариантные к информационному содержанию файла. Уже сейчас можно браться за разработку приборных методов и программных средств, обнаруживающих факт применения стеганографических программ, имеющихся на рынке программного продукта. Незавершенность заключается в одномерности критериев. Не используется априорная информация, содержащаяся в конкретном файле и его динамике. Необходим переход к многомерным статистикам, формированию и распознаванию образов, предсказанию траектории развития аудио- и видеопроцесса (видеосигнала) с достаточной точностью.

Криптографический примитив MV2

Ю. В. Виланский

Введение

В данной работе описывается новый криптографический примитив MV2, который создает криптотекст состоящий из двух частей. По существу он является симметричным, итерационным, вероятностным шифром, каждая итерация которого напоминает раунд подстановочно-перестановочной сети, при выполнении которой обрабатывается весь текст, а не блок, как в блочных шифрах. Отметим, что в MV2 на вход раунда поступает текст произвольной длины, а на выходе получаем два текста. Механизм разделения основан на применении специальных ключевых подстановочных преобразований. Задавая параметры, можно управлять длинами полученных текстов. MV2 не относится ни к классу блочных, ни к классу поточных шифров.

Одним из рекомендуемых нами режимов использования является каскад — поточный шифр — MV2. Исходный текст поступает на вход первой компоненты каскада, которая является поточным шифром.

Криптографический примитив MV2 позволяет реализовать некоторые новые технологии (см. [1]).

Отображения с выходами переменной длины

В криптографическом примитиве MV2 используются подстановочные преобразования вида:

$$T\colon \{0,1\}^n \to \bigcup_{i=r+1}^{n-1} \big\{\{0,1\}^i \times \{n-i\}\big\} \bigcup \big\{\{0,1\}^r \times \{n-r, n-r+1\}\big\}. \quad (1)$$

Преобразование вида (1) можно представить парой функций $T = (c,\ f)$. Здесь $c\colon \{0,1\}^n \to \bigcup_{i=r}^{n-1}\{0,1\}^i$ — отображение двоичной строки длиной n бит в двоичную строку меньшей длины, причем для образов $y \in \bigcup_{i=r+1}^{n-1}\{0,1\}^i$ существует единственный прообраз, а для образов $y \in \{0,1\}^r$ существует ровно 2 прообраза. Вторая функция $f\colon \{0,1\}^n \to \{1, 2, \ldots, n-r+1\}$ — отображает множество двоичных векторов длиной n в множество $n-r+1$ чисел. При этом функции c и f связаны между собой следующим образом — если образ $y = c(x)$, такой, что $y \in \{0,1\}^k$, $r+1 \leqslant k \leqslant n-1$, то $f(x) = n-k$, а если образ $c(x) = y \in \{0,1\}^r$, то $f(x)$ принимает либо значение $n-r$ либо $n-r+1$. Множество таких преобразований будем обозначать $\mathcal{F}_n^r$. Информационные утечки для похожих преобразований изучались в [2].

Значения выходов $z = f(x)$ преобразования $T = (c,\ f) \in \mathcal{F}_n^r$, можно закодировать двоичным (ДК) следующим образом:

z	1	2	3	…	$n-r$	$n-r+1$
ДК	1	01	$0^2 1$	…	$0^{n-r-1}1$	0^{n-r}

где 0^i обозначает битовую строку из i нулей. При равномерном распределении входов такой код будет совпадать с кодом Хаффмана.

При таком представлении выходов, преобразования вида (1) отображают n-битную строку x в пару $(c,\ f)$, состоящую из двух строк переменной длины.

В MV2, исходный текст M поступающий на вход подстановочного преобразования $T = (c,\ f)$ представляется, как конкатенация $M = x_1 \,\|\, x_2 \,\|\, \ldots \,\|\, x_L$ символов $x_i \in \{0,1\}^n$. Под $c(M)$ понимается двоичная строка, которая получается конкатенацией образов $c(x_i)$, а под $f(M)$ — двоичная строка, представляющая конкатинацию соответствующих двоичных кодов образов $f(x_i)$

$$c(M) = c(x_1) \,\|\, c(x_2) \,\ldots\, \|\, c(x_L); \qquad f(M) = f(x_1) \,\|\, f(x_2) \,\ldots\, \|\, f(x_L). \quad (2)$$

Двоичную последовательность $c(M)$ будем соответственно называть остатком, а $f(M)$ — флагами текста M при выполнении преобразования $T = (c,\ f)$.

Пусть задано некоторое $T = (c,\ f) \in \mathcal{F}_n^r$, тогда для любой двоичной последовательности $C \in \{0,1\}^L$, мощность $N_c = \#\{M : c(M) = C\}$ будет равна [3, с. 215]

$$N_c \geqslant \sum_{l=[L/(n-1)]}^{[L/r]} \sum_{k=0}^{[(L-rl)/(n-r)]} (-1)^k \binom{l}{k} \binom{L-(r-1)l-(n-r)k-1}{l-1}. \quad (3)$$

Утверждение 1. *Пусть задано $T = (c,\ f) \in \mathcal{F}_n^r$ и текст $M = x_1 \,\|\, \ldots \,\|\, x_L$, из L символов x_i, которые случайно и равномерно выбираются из $\{0,1\}^n$. Тогда для математического ожидания длины остатка $\mathsf{E}(|c(M)|)$ и математического ожидания длины флагов $\mathsf{E}(|f(M)|)$ выполняется*

$$\mathsf{E}(|c(M)|) = \left(n - 2 + 2^{r-n+1}\right) \cdot L; \qquad \mathsf{E}(|f(M)|) = \left(2 - 2^{r-n+1}\right) \cdot L. \quad (4)$$

В базовой реализации MV2, $n = 8$ и $r = 3$, т. е. каждый байт входного текста отображается в пару образов, один из которых (остаток), имеет длину от 3 до 7 бит, а второй (флаг) представляет собой код значения от 1 до 6.

Свойства отображений вида (1) можно найти в [4].

Криптографический примитив MV2

Опишем входные и выходные параметры MV2:

Зашифрование

Вход:	Исходный текст	M $(8 \times l$ bits$)$
	Секретный ключ	K $(\leqslant 32 \times 8 \times 260$ bits$)$
	Число раундов	P
Выход:	Шифротекст	(C, F)

Расшифрование

Вход:	Шифротекст	(C, F)
	Секретный ключ	K $(\leqslant 32 \times 8 \times 260$ bits$)$
Выход:	Сообщение об ошибке или исходный текст	M $(8 \times l$ bits$)$

Выход C мы называем *ядром*, в выход F — *флагами*.

Устройство для реализации метода на основании которого построен криптографический примитив MV2 описано в [7].

Весь процесс зашифрования, осуществляемый MV2 можно разбить на раунды. Каждый раунд состоит из *линейного слоя* и *нелинейного слоя*.

Линейный слой реализуется перестановочным преобразованием которое обеспечивает высокую степень локальной диффузии. Для осуществления нелинейных преобразований используются отображения вида (1), которые задаются с помощью секретных таблиц, являющихся ключевой информацией.

Зашифрование выполняется по следующему алгоритму:

BEGIN
$C_0 = M$; $F = \varnothing$;

For $i = 1$; **To** $i = P$

1. $C_{i-1} = \mathbf{Mix}(C_{i-1})$ — остаток предыдущего раунда перемешивается поблочно;
2. Генерируется случайное(псевдослучайное) число R_i;
3. Используя R_i, вычисляется j — номер подстановочного преобразования из ключа K;
4. $T_j(C_{i-1}) = (C_i, F_i)$ — выполняется подстановочное преобразование T_j в результате чего образуется новый остаток C_i и флаги F_i;
5. $C_i = R_i \| C_i$; $F = F_i \| F$;

End.

$C = C_P$;
END.

В MV2, число раундов может быть задано явно или косвенно. При косвенном задании, указывается верхняя граница L_c и число раундов P это номер раунда на котором длина выходных данных C_i стала меньше L_c.

Число раундов влияет на стойкость MV2 [5].

Оценку стойкости шифров принято выполнять либо построением атак либо по косвенным признакам. При этом в традиционных шифрах хотя бы криптотекст считается известным. Для алгоритма MV2, необходимо рассмотреть и другие варианты например — известно только ядро или ядро и ключи, только флаги или флаги и ключи.

Если известно только ядро и нет ограничения на количество раундов, то даже при известных ключах, имеется бесконечное множество текстов которые дают такое ядро. Оценки для других случаев можно найти в [4], [5].

Для современных шифров очень редко можно построить аналитическую модель позволяющую оценить их стойкость. Обычно используются некоторые критерии стойкости и выполняются тесты для оценки соответствия алгоритма этим критериям. Для подстановочно-перестановочных сетей обычно используют критерии зависимости и статистические тесты. Для базовой реализации MV2 проводилось тестирование на соответствие критериям зависимости. Для тестировании была разработана модель, основанная на определениях и моделях, которые использовались для тестов финалистов AES [6]. Поскольку выходы MV2 представляют собой двоичные строки различной длины, то для построения моделей на множестве $\bigcup_{i=1}^{\infty}\{0, 1\}^i$ вводилась следующая метрика, обобщающая расстояние Хемминга:

Определение 1. Расстоянием между двумя двоичными строками $y_1 \in \{0, 1\}^i$ и $y_2 \in \{0, 1\}^j$, где

$$y_1 = (Y_{11}, Y_{12}, \ldots, Y_{1i}), \quad y_2 = (Y_{21}, Y_{22}, \ldots, Y_{2j}), \quad Y_{ks} \in \{0, 1\},$$

называется величина

$$\widetilde{\mathrm{wt}}(y_1, y_2) = \mathrm{wt}((Y_{11}, \ldots, Y_{1l}), (Y_{21}, \ldots, Y_{2l}) + m - l, \qquad (5)$$

где $\mathrm{wt}(\cdot)$ — обычное расстояние Хемминга, $l = \min\{i, j\}$ и $m = \max\{i, j\}$.

Результаты тестирования позволяют утверждать, что MV2 удовлетворяет критериям зависимости и предположить, что стойкость MV2 возрастает при увеличении длины входного текста.

Более подробные результаты исследования MV2 можно найти в работах [4], [5].

Заключение

Описанный криптографический примитив обладает рядом отличительных особенностей:

- криптотекст состоит из двух частей — ядра и флагов;
- для восстановления исходного текста необходимо обладать всеми тремя компонентами — ключом, ядром и флагами;
- не зависимо от размера и содержания исходного текста, ядро можно сделать достаточно малым (но не меньше некоторой, зависящей от реализации величины);
- длина ключа, фактически примененного для шифрования, пропорциональна количеству раундов преобразования;
- алгоритм MV2 — псевдослучайный, при многократном зашифровании одного и того же исходного текста получаются различные пары ядер и флагов, что позволяет не менять ключи при длительном их использовании;
- криптографический примитив MV2 позволяет распараллелить процесс шифрования;
- криптографический примитив MV2 может быть легко доработан до варианта обеспечивающего аутентификацию сообщения.

Список литературы

[1] *Мищенко В. А.* Криптография и массовые технологии современного рынка // Успехи современного естествознания. 2004. № 5. Приложение № 1. С. 146–149.

[2] *Виланский Ю. В.*, *Лепин В. В.* Информационные утечки в отображениях с образами различной длины // Весці НАН Беларусі. Сер. фіз.-мат. навук. 2004. № 3. С. 47–53.

[3] *Сачков В. Н.* Введение в комбинаторные методы дискретной математики. М.: Наука, 1982 г.

[4] *Виланский Ю. В.*, *Лепин В. В.*, *Мищенко В. А.* Двухканальный алгоритм шифрования MV2 // Вести Института современных знаний. № 3–4. 2003. С. 113–121.

[5] *Виланский Ю. В.*, *Лепин В. В.*, *Мищенко В. А.* Двухканальный алгоритм шифрования MV2 (продолжение) // Вести Института современных знаний. 2004, № 1. С. 77–88.

[6] *Preneel B.*, *Bosselaers A.*, *Rijmen V.*, *Van Rompay B. Granboulan L.*, *Stern J.*, *Murphy S.*, *Dichtl M.*, *Serf P. Biham E.*, *Dunkelman O.*, *Furman V. Koeune F.*, *Piret G.*, *Quisquater J-J.*, *Knudsen L.*, *Raddum H.* “Comments by the NESSIE Project on the AES Finalists” [Electronic resource]. 2000. `http://csrc.nist.gov/CryptoToolkit/aes/round2/comments/20000524-bpreneel.pdf`.

[7] *Mischenko V. A.*, *Zakharau U. U.*, *Vilansky Y. V.*, *Verzhbalovich D. I.* Method for encrypting information and device for realization of the method // International Publication Number: WO 00/65767. `http://pctgazette.wipo.int/`. 23 p.

Теоретико-групповая характеризация неавтономных линейных регистров сдвига над свободным модулем

О. А. Козлитин

Пусть R — конечное коммутативное кольцо с единицей, ${}_RM$ — свободный модуль ранга m, $n \in \mathbb{N}$, $f\colon M^n \to M$, $f(0, 0, \ldots, 0) = 0$. *Неавтономным регистром сдвига* R_n^f назовем автомат (I, S, δ), где $I = M$, $S = M^{(n)}$, а функция перехода δ определена равенством

$$\delta(i, s) = \delta_i(s) = \delta_i \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} x_2 \\ \vdots \\ x_n \\ f(x_1, \ldots, x_n) + i \end{pmatrix}. \qquad (1)$$

Функцию f из равенства (1) назовем *функцией обратной связи*. Регистр с линейной функцией обратной связи называется *линейным*. Вместо слов «неавтономный линейный регистр сдвига» будем в дальнейшем писать «НЛРС».

Автомат R_n^f регулярен тогда и только тогда, когда функция

$$f(x_1, x_2, \ldots, x_n)$$

биективна по переменной x_1. В данной статье рассматриваются только регулярные регистры.

В регулярном регистре функции δ_i $(i \in I)$ суть подстановки из симметрической группы $S(M^{(n)})$. Для удобства выкладок будем считать, что групповая операция в $S(M^{(n)})$ — не произведение подстановок, а их композиция. Обозначим через G^f группу $\langle \delta_i \mid i \in I \rangle < S(M^{(n)})$ и назовем ее *группой регистра* R_n^f. Для каждого $a \in I$ обозначим через $\left[\delta_a \delta_0^{-1}\right]$ множество подстановок, сопряженных с подстановкой $\delta_a \delta_0^{-1}$ в группе G^f. Обозначим через B_f группу $\left\langle \bigcup_{a \in I} \left[\delta_a \delta_0^{-1}\right] \right\rangle$ и назовем ее *базой регистра* R_n^f. Можно показать, что база B_f — транзитивный нормальный делитель группы G^f.

Работа выполнена при поддержке гранта Президента РФ № НШ-2358.2003.9.

Далее если x есть строка (столбец) над модулем M, то x^t есть результат *транспонирования* x. Через e_k $(k \in \overline{1, n})$ обозначим отображение, переводящее элемент $i \in M$ в столбец $(0, 0, \ldots, i, \ldots, 0)^t \in M^{(n)}$, где элемент i стоит на k-ом месте, а через $e_k i$ — результат применения отображения e_k к элементу $i \in M$. Пусть $F\colon S \to S$ есть некоторое отображение, и $F(s) = \sum_{k=1}^n e_k y_k(s)$. Будем называть функции y_k $(k \in \overline{1, n})$ *координатными функциями* отображения F.

Для всякого $a \in M^{(n)}$ обозначим через t_a сдвиг на a, то есть преобразование $s \mapsto s + a$. Тогда для всякого $i \in I$ $\delta_i = t_{e_n i} \delta_0$, откуда $t_{e_n i} = \delta_i \delta_0^{-1}$. Поскольку

$$\delta_i^{-1} = \delta_0^{-1} t_{-e_n i},$$

база B_f регистра R_n^f равна $\left\langle \tau_{k,i} \mid k \in \overline{0, \operatorname{ord} \delta_0 - 1},\ i \in I \right\rangle$, где

$$\tau_{k,i} = \delta_0^k t_{e_n i} \delta_0^{-k}.$$

Обозначим через Σ группу сдвигов $\left\{ t_a \mid a \in M^{(n)} \right\} < S(M^{(n)})$. Будем говорить, что функция f *аддитивна*, если $f(x + y) = f(x) + f(y)$ для любых $x, y \in S$.

Утверждение 1. *Функция f аддитивна если и только если $B_f = \Sigma$.*

Доказательство. Необходимость. Покажем, что для любого $a \in M^{(n)}$ существует $b \in M^{(n)}$ такой, что $\delta_0^{-1} t_a \delta_0 = t_b$. Действительно, если $a = (a_2, \ldots, a_n, a_1)^t$, то

$$\delta_0^{-1} t_a \delta_0 \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix} = \begin{pmatrix} f_1^{-1}(f(x_1, \ldots, x_n) + a_1, x_2 + a_2, \ldots, x_n + a_n) \\ x_2 + a_2 \\ \vdots \\ x_n + a_n \end{pmatrix}, \qquad (2)$$

где f_1^{-1} есть функция, обратная к функции f по переменной x_1:

$$f_1^{-1}(f(x_1, x_2, \ldots, x_n), x_2, \ldots, x_n) = f(f_1^{-1}(x_1, x_2, \ldots, x_n), x_2, \ldots, x_n) = x_1.$$

Обозначим через $g(x)$ первую координатную функцию преобразования $\delta_0^{-1} t_a \delta_0$. Имеем:

$$f_1^{-1}(f(s) + a_1, x_2 + a_2, \ldots, x_n + a_n) = g(s),$$
$$f(g(s), x_2 + a_2, \ldots, x_n + a_n) = f(s) + a_1.$$

Вычтем $f(s)$ из обеих частей последнего равенства. Пользуясь аддитивностью f, получаем

$$f(g(s) - x_1, a_2, \ldots, a_n) = a_1.$$

Если $b_1 = f_1^{-1}(a_1, a_2, \ldots, a_n)$, то $g(s) = x_1 + b_1$, т. е. $\delta_0^{-1} t_a \delta_0 = t_b$, где $b = (b_1, a_2, \ldots, a_n)^t$. Поэтому все подстановки $\tau_{k,i}$ суть сдвиги. Отсюда $B_f \leqslant \Sigma$. Группа Σ регулярна, поэтому из транзитивности B_f следует равенство $B_f = \Sigma$.

Достаточность. Если $B_f = \Sigma$, то для любого $a \in M^{(n)}$ существует $b \in M^{(n)}$ такой, что $\delta_0^{-1} t_a \delta_0 = t_b$. Пусть $a = (a_2, \ldots, a_n, a_1)^t$ — произвольный вектор из множества $M^{(n)}$, и первая координата соответствующего вектора b есть b_1. Из равенства (2) имеем:

$$f_1^{-1}(f(s) + a_1, x_2 + a_2, \ldots, x_n + a_n) = x_1 + b_1,$$

откуда

$$f(x_1 + b_1, x_2 + a_2, \ldots, x_n + a_n) = f(s) + a_1. \tag{3}$$

Последнее равенство выполняется для любого $s \in M^{(n)}$. Положив $s = 0$, имеем

$$f(b_1, a_2, \ldots, a_n) = a_1. \tag{4}$$

Выберем и зафиксируем $y = (y_1, y_2, \ldots, y_n) \in M^{(n)}$. Положим в последнем равенстве $a_1 = f(y_1, y_2, \ldots, y_n)$, $a_i = y_i$ для всех $i \geqslant 2$. Тогда в силу биективности функции f по первой переменной имеем: $(b_1, a_2, \ldots, a_n) = (y_1, y_2, \ldots, y_n)$. Из равенств (3) и (4) получаем

$$f(s + y) = f(s) + f(y)$$

для любых s, $y \in M^{(n)}$. Таким образом, функция f аддитивна.

Утверждение доказано. □

В работе [2] показано, что если $R = M = \mathbb{Z}_2$, то единственной абелевой группой в классе всех баз является группа сдвигов. Таким образом, двоичный регистр линеен тогда и только тогда, когда его база абелева.

Следующая теорема обобщает центральный результат работы [2]:

Теорема 1. *Если база регистра R_n^f абелева, то существуют такие преобразования A_1, A_2, …, A_n модуля ${}_RM$, что*

$$f(x_1, x_2, \ldots, x_n) = A_n(x_1) + A_{n-1}(x_2) + \ldots + A_1(x_n),$$

причем для всех $k \in \overline{1, n}$ $A_k(0) = 0$.

К сожалению, объем данной статьи не позволяет привести здесь доказательство. Поэтому теорема 1 и все последующие результаты сформулированы без обоснования.

Для каждого $k \in \overline{1, n}$ обозначим через f_k функцию

$$f(0, \ldots, 0, x_k, 0, \ldots, 0),$$

где нули стоят на всех местах кроме k-го. Будем говорить, что функция над конечным полем *лишена свободных квадратов*, если в ее представлении приведенным многочленом каждый моном, существенно зависящий от одной переменной, линеен. Справедливо

Следствие 1. *Во введенных обозначениях функция f линейна (аддитивна) тогда и только тогда, когда база B_f абелева и все функции f_1, f_2, …, f_n линейны (аддитивны). В частности, если $R = M = \mathrm{GF}(q)$ и функция f лишена свободных квадратов, то регистр R_n^f линеен тогда и только тогда, когда его база абелева.*

Регистр, у которого все функции f_1, f_2, …, f_n линейны, будем называть *локально-линейным*. Пусть $R = \mathrm{GR}(q^d, p^d)$ — кольцо Галуа, а многочлен $G(x) \in R[x]$ является реверсивным и унитарным. Согласно [1] *периодом* $G(x)$ называется минимальное натуральное t со свойством $G(x) \mid x^t - 1$. Обозначим период $G(x)$ через $T(G)$, а результат покоэффициентного приведения $G(x)$ по модулю радикала через $\overline{G}$. Назовем $G(x)$ *сепарабельным*, если $\overline{G}$ не имеет кратных корней в своем поле разложения, и *отмеченным*, если $T(G) = T(\overline{G})$.

Рассмотрим линейный регистр R_n^f над кольцом R. Если

$$f(x_1, x_2, \ldots, x_n) = a_n x_1 + a_{n-1} x_2 + \ldots + a_1 x_n,$$

назовем *характеристическим многочленом* регистра R_n^f многочлен

$$\chi_f(x) = x^n - a_1 x^{n-1} - a_2 x^{n-2} - \ldots - a_n \in R[x].$$

Пусть G — группа, $N \lhd G$, $H < G$. В случае, если определено полупрямое произведение группы N на группу H, будем обозначать его через $N \lambda H$. Справедливы следующие утверждения, обобщающие соответственно предложение 4.1 и теорему 3.1 из работы [2]:

Утверждение 2. *Если $R = \mathrm{GR}(q^d, p^d)$, R_n^f — локально-линейный регистр над кольцом R, то следующие условия эквивалентны:*

1. *R_n^f — линейный регистр с сепарабельным отмеченным характеристическим многочленом.*
2. *$G^f = N \leftthreetimes \langle g \rangle$, где N — абелев нормальный делитель и* ord g *не делится на p.*
3. *G^f содержит абелев нормальный делитель, индекс которого не делится на p.*

Теорема 2. *Пусть R — конечное коммутативное кольцо характеристики 2. Тогда для локально-линейного регистра R_n^f над модулем ${}_RM$ следующие утверждения эквивалентны:*

1. *R_n^f — НЛРС,*
2. *$G^f = N \leftthreetimes \langle g \rangle$, где N — элементарная абелева 2-группа,*
3. *G^f есть расширение элементарной абелевой 2-группы с помощью циклической группы.*

Неприводимый унитарный многочлен $G(x) \in \mathrm{GF}(q)[x]$ называется *примитивным*, если его корни суть примитивные элементы поля разложения. Следующие утверждения обобщают соответственно предложения 4.2, 4.4 и 4.5 из работы [2]:

Утверждение 3. *Для локально-линейного регистра R_n^f над полем $\mathrm{GF}(q)$ следующие утверждения эквивалентны*:

1. *R_n^f есть линейный регистр сдвига с примитивным характеристическим многочленом.*
2. *G^f есть 2-транзитивная группа, у которой каждый неединичный элемент оставляет на месте не более одного символа алфавита состояний.*
3. *G^f есть точно 2-транзитивная группа.*

Утверждение 4. *Пусть $R = M = \mathrm{GF}(q)$, $n > 1$, регистр R_n^f — локально-линейный. Тогда*

1) *если G^f — примитивная двухступенно разрешимая группа подстановок множества S, то R_n^f есть НЛРС с неприводимым характеристическим многочленом*;
2) *если дополнительно известно, что $R = M = \mathbb{Z}_p$, верно и обратное утверждение.*

Утверждение 5. *Пусть $R = M = \mathrm{GF}(q)$, регистр R_n^f — локально-линейный. Тогда*

1) *если G^f примитивна и любой ее неединичный элемент оставляет на месте не более одного символа, то R_n^f есть НЛРС с неприводимым характеристическим многочленом*;
2) *если дополнительно известно, что $R = M = \mathbb{Z}_p$, верно и обратное утверждение.*

Напомним [3], что транзитивная группа подстановок называется *группой Фробениуса*, если любой ее неединичный элемент оставляет на месте не более одного символа. Множество подстановок группы Фробениуса, лишенных неподвижных точек, образуют вместе с единичной подстановкой подгруппу, называемую *ядром* группы Фробениуса. Согласно классической теореме Фробениуса ядро группы Фробениуса есть ее регулярный нормальный делитель. Группа подстановок *характеристически проста*, если она лишена собственных характеристических подгрупп, то есть подгрупп, инвариантных относительно всех автоморфизмов исходной группы. Кроме того, согласно [1] *порядком* многочлена над конечным полем называют наименьшее общее кратное мультипликативных порядков его корней в поле разложения.

Имеет место следующий критерий:

Теорема 3. *Пусть $R = M = \mathbb{Z}_p$, f лишена свободных квадратов. Тогда следующие утверждения эквивалентны*:

1. *R_n^f есть НЛРС, его характеристический многочлен χ_f сепарабелен, и порядок χ_f совпадает с порядком любого его корня в поле разложения.*
2. *G^f есть группа Фробениуса, ядро которой есть группа сдвигов.*
3. *G^f есть группа Фробениуса с характеристически простым ядром.*
4. *G^f есть группа Фробениуса, ядро которой — элементарная абелева p-группа.*
5. *G^f есть группа Фробениуса с абелевым ядром.*

Следствие 2. *Пусть R_n^f — локально-линейный регистр над $\mathbb{Z}_p$ и n — простое число. Тогда R_n^f — линейный с неприводимым характеристическим многочленом если и только если выполняется любое из условий* (2)–(5) *теоремы* 3.

Автор выражает признательность А. А. Нечаеву за постановку задачи и обсуждение полученных результатов.

Список литературы

[1] *Глухов М. М.*, *Елизаров В. П.*, *Нечаев А. А.* Алгебра. М.: Гелиос АРВ, 2003.

[2] *Башев В. А.* Теоретико-групповая характеризация неавтономных линейных регистров сдвига. Труды по дискретной математике, т. 7, 2004.

[3] *Холл М.* Теория групп. М.: изд. иностр. лит., 1962.

О фрагментах наборов чисел, полученных приведением по модулю два перестановок с ограниченным числом инверсий

Д. А. Куропаткин

Пусть зафиксированы натуральные числа n, m и d. Натуральные числа k и r, $0 \leqslant r < d$, таковы, что $n = kd + r$.

Множество перестановок чисел $1, 2, \ldots, n$ обозначим через $S^{(n)}$. В соответствии с [1] числа u и v, $1 \leqslant u < v \leqslant n$, образуют инверсию в перестановке $\delta = (\delta_1, \delta_2, \ldots, \delta_n)$ из $S^{(n)}$, если существуют натуральные числа i и j такие, что $\delta_i = u$ и $\delta_j = v$, а $i > j$. Здесь рассматриваем только инверсии, которые образованы числами u и $u+1$, $1 \leqslant u \leqslant n-1$. Множество перестановок из $S^{(n)}$, число инверсий в каждой из которых не превосходит m, обозначим через $S_m^{(n)}$.

Пусть $S_{k,r}^{(l)}(d)$ — множество наборов $\gamma = \gamma_1\gamma_2\ldots\gamma_l$ чисел от 1 до d, в каждом из которых любое число из множества $\{1, 2, \ldots, r\}$ встречается не более $k+1$ раз, а любое число из множества $\{r+1, r+2, \ldots, d\}$ встречается не более k раз.

Определим отображение $F\colon S^{(n)} \to S_{k,r}^{(n)}(d)$ следующим образом: если $\delta = \delta_1\delta_2\ldots\delta_n \in S^{(n)}$, то набор $F(\delta) = \gamma = \gamma_1\gamma_2\ldots\gamma_n$ получен из перестановки δ путем приведения каждого из его элементов по модулю d, то есть $\gamma_i = \delta_i \pmod d$, $i = 1, \ldots, n$. При этом берется наименьший положительный вычет.

Множество $F(S_m^{(n)})$ обозначим через $A(n, m, d)$.

Шаблоном назовем набор чисел $(i_1, i_2, \ldots, i_l)$, $1 \leqslant l < n$, из множества $\{1, 2, \ldots, n\}$ такой, что $1 \leqslant i_1 < i_2 < \ldots < i_l \leqslant n$. Набор чисел $\tau \in S_{k,r}^{(l)}(d)$ назовем допустимым для шаблона $(i_1, i_2, \ldots, i_l)$, если существует такой набор чисел $\gamma \in A(n, m, d)$, что $\gamma_{i_j} = \tau_j$ для всех $j = 1, 2, \ldots, l$. Множество наборов, допустимых для шаблона $(i_1, i_2, \ldots, i_l)$, обозначим $\mathrm{PosSel}(i_1, i_2, \ldots, i_l)$.

Пусть $d = 2$ и $i \in \{1, 2\}$, $j \in \{1, 2, \ldots, l\}$. Для произвольного набора $\tau \in S_{k,r}^{(l)}(2)$ обозначим через $N_i(j, \tau)$ количество чисел набора τ равных i и расположенных на местах с 1 по j включительно.

Введем обозначения

$$Y(j, \tau) = N_2(j, \tau) - N_1(j, \tau), \quad Y(0, \tau) = 0,$$
$$Y_{\max}(\tau) = \max_{j=0,\ldots,l} Y(j, \tau), \quad Y_{\min}(\tau) = \Big|\min_{j=0,\ldots,l} Y(j, \tau)\Big|.$$

Через $I(A)$ обозначим индикатор события A, то есть

$$I(A) = \begin{cases} 1, & \text{если событие } A \text{ имеет место;} \\ 0, & \text{иначе.} \end{cases}$$

Теорема. *Пусть* $\tau \in S_{k,r}^{(l)}(2)$, *где* $1 \leqslant l < n$. *Тогда*

1) *набор* τ *принадлежит множеству* $\mathrm{PosSel}(1, 2, \ldots, l)$ *тогда и только тогда, когда*

$$Y_{\max}(\tau) + Y_{\min}(\tau) - I(A) \leqslant m,$$

где $A = (Y_{\min}(\tau) \neq 0) \vee (Y(l, \tau) \leqslant 2(k-1) - l - r)$;

2) *набор* τ *принадлежит множеству*

$$\mathrm{PosSel}(n-l+1, n-l+2, \ldots, n)$$

тогда и только тогда, когда

$$Y_{\max}(\tau) + Y_{\min}(\tau) - I(A) \leqslant m,$$

где

$$A = (Y(l, \tau) \geqslant r) \vee (Y_{\min}(\tau) + Y(l, \tau) > r) \vee \vee (l + r - Y(l, \tau) \leqslant 2(k-1)).$$

Данная работа является продолжением результатов, изложенных в [3]. В отличие от настоящей работы, где рассматриваются части наборов определенного вида, в тезисах [3] исследовались сами наборы указанного вида.

Список литературы

[1] *Глухов М. М.*, *Елизаров В. П.*, *Нечаев А. А.* Алгебра. М.: Гелиос АРВ, 2003.

[2] *Сачков В. Н.* Введение в комбинаторные методы дискретной математики. М.: Наука, 1982.

[3] *Куропаткин Д. А.* Количество наборов чисел, полученных приведением по модулю два перестановок с ограниченным числом инверсий. Обозрение прикл. и промышл. матем., 2004, т. 11, в. 1, с. 123–124.

О числе ненулевых элементов треугольника Паскаля над конечным полем

Е. В. Кутырева

Рассмотрим аналог треугольника Паскаля T_s, состоящий из s строк элементов произвольного конечного поля $\mathrm{GF}(q)$, $q = p^l$. В этом треугольнике элементы i-й строки $\left(x_1^{(i)}, x_2^{(i)}, \ldots, x_i^{(i)}\right)$, $i = \overline{1,\, s-1}$, получаются из элементов $(i+1)$-й строки $\left(x_1^{(i+1)}, x_2^{(i+1)}, \ldots, x_{i+1}^{(i+1)}\right)$ по следующему правилу: $x_j^{(i)} = \alpha x_j^{(i+1)} + \beta x_{j+1}^{(i+1)}$, $j = \overline{1,\, i}$, где $\alpha,\, \beta \in \mathrm{GF}(q)^*$. Последняя s-я строка может быть произвольной. Число ненулевых элементов в треугольнике T_s обозначим через ξ. Через s_0 обозначим размер максимального треугольника T_{s_0}, состоящего целиком из нулевых элементов и содержащегося в треугольнике T_s.

Обозначим через $r \geqslant 0$ увеличенное на единицу количество строк между треугольником T_{s_0} и последней s-й строкой и рассмотрим трапецию Tr, образованную данными r строками (вместе с s-й строкой). В трапецию Tr не включаем элементы вне угла, образованного двумя сторонами треугольника T_{s_0}, непараллельными основаниям трапеции. Строку, следующую за основанием треугольника T_{s_0}, будем считать 1-й строкой трапеции Tr.

Разобъем строки трапеции Tr на $\lceil \log_p r \rceil + 1$ групп по $(p-1)p^{j-1}$ строк, где j — номер группы, $j \in \overline{1,\, \lceil \log_p r \rceil}$. Группа с номером 0 состоит из 1-й строки трапеции Tr. Последняя группа может быть неполной.

Пусть порядок элемента $-\alpha\beta^{-1}$ равен c. Нетрудно убедиться, что верно следующее

Утверждение 1. *Период строки трапеции* Tr *из группы с номером j равен cp^j, $j \in \overline{0,\, \lceil \log_p r \rceil}$.*

Также верно

Утверждение 2. *На периоде любой строки трапеции* Tr *содержится не менее c ненулевых элементов.*

С помощью вышеприведенных утверждений доказана следующая

Теорема. *Для любого $k > 0$ при достаточно большом $s > S$ в случае $\xi \leqslant ks$ имеем $s - s_0 \leqslant R$, где $R = R(k)$ и $S = S(k)$ — зависящие от k константы.*

Данная теорема для случая $q = 2$ приведена в [1]. Из сформулированной теоремы вытекает

Следствие. *Имеется монотонная неограниченная последовательность рациональных чисел*

$$0 = k_0 < k_1 < k_2 < \ldots$$

такая, что для любого фиксированного $K > 0$ при $\xi \leqslant Ks$ и достаточно больших $s > S(K)$ имеем: $M_s(\xi) = 0$, если

$$\xi \notin \bigcup_{i=0}^{\infty} (k_i s - \varepsilon_i,\, k_i s + \varepsilon_i),$$

где ε_i, $i \geqslant 0$ — некоторые неотрицательные константы.

Таким образом, при наличии относительно небольшого числа ненулевых элементов в треугольнике T_s $\xi \leqslant Ks$ величина ξ может принимать значения только из интервалов $(k_i s - \varepsilon_i,\, k_i s + \varepsilon_i)$, $i \geqslant 0$, для любых достаточно больших s.

Список литературы

[1] *Малышев Ф. М., Кутырева Е. В.* Об одном свойстве булевых аналогов треугольника Паскаля. Обозрение прикладной и промышленной математики. Т. 11. Вып. 2. С. 245–246. М.: ТВП, 2004.

Задачи нахождения оценок параметров распределения слагаемого по наблюдениям суммы случайных величин на конечной абелевой группе

А. В. Лапшин

Пусть G — конечная абелева аддитивная группа, α, β_1, …, β_s — случайные величины на группе G. Пусть вероятности $\pi_j(g)=\Pr\{\beta_j=g\}$, $g\in G$, $j=1,\ldots,s$ известны, а $p(g)=\Pr\{\alpha=g\}$, $g\in G$, неизвестны и их требуется оценить по результатам испытаний над случайными величинами $\alpha+\beta_j$, $j=1,\ldots,s$. Полагаем, что случайные величины α, β_1, …, β_s независимы в совокупности, а число испытаний над случайной величиной $\alpha+\beta_j$ равно n_j, $j=1,\ldots,s$.

В качестве оценки $p^*(g)$, $g\in G$, распределения вероятностей $p(g)$, $g\in G$, рассмотрим оценку $p^*(g)$, $g\in G$, соответствующую

$$\min_{p(g)}\sum_{j=1}^{s}\|p*\pi_j(g)-P_j^*(g)\|^2, \tag{1}$$

где:

- $p*\pi_j(g)$ — свертка распределений $p(g)$ и $\pi_j(g)$;
- $P_j^*(g)=n_j(g)/n_j$ — относительная частота исхода g в n наблюдениях;
- $n_j(g)$ — число испытаний, в которых реализация случайной величины $\alpha+\beta_j$ равна g, $\sum_g n_j(g)=n_j$, $g\in G$, $j=1,\ldots,s$;
- $\|f(g)\|^2=\sum_{g\in G}|f(g)|^2$, $f(g)$ — произвольная комплекснозначная функция переменной g.

Точность оценки $p^*(g)$ распределения вероятностей $p(g)$, $g\in G$, будем характеризовать величиной

$$\delta=\|p^*-p\|^2. \tag{2}$$

Найдем среднее значение величины δ.

Теорема 1. *Пусть* $\sum_{j=1}^{s}|\widehat{\pi}_j(\chi)|^2\neq 0$, $\chi\in\widehat{G}$, *тогда для случайной величины* δ *имеет место соотношение*

$$\mathsf{E}\delta=\frac{1}{|G|}\sum_j\frac{1}{n_j}\sum_{\chi\neq 1}\frac{|\widehat{\pi}_j(\chi)|^2\left(1-|\widehat{p}(\chi)|^2|\widehat{\pi}_j(\chi)|^2\right)}{\left(\sum_{j'}|\widehat{\pi}_{j'}(\chi)|^2\right)^2}, \tag{3}$$

где $\widehat{p}(\chi)$ *— характеристическая функция произвольного распределения вероятностей* $p(g)$ *на группе* G, *а* $\widehat{G}$ *— двойственная группа группе* G (*т.е. группа характеров группы* G).

Рассмотрим задачу оценки распределения слагаемого по сумме случайных величин на конечной абелевой группе в случае, когда функции $\pi_j(g)$, $g\in G$, $j=1,\ldots,s$, неизвестны. При этом будем предполагать, что известны относительные частоты $\pi_j^*(g)$ появления исходов $g\in G$ в N_j независимых испытаниях над случайной величиной β_j с распределением вероятностей $\pi_j(g)$, $g\in G$, $j=1,\ldots,s$, причем эти испытания независимы от n_j испытаний над случайной величиной $\alpha+\beta_j$, в которых случайная величина β_j присутствует в качестве одного из слагаемых. В рассматриваемом случае кроме величин $P_j^*(g)$ являются случайными и величины $\pi_j^*(g)$, и, соответственно, величины $\widehat{P}_j^*(\chi)$ и $\widehat{\pi}_j^*(\chi)$ также являются случайными. По аналогии с предыдущим случаем рассмотрим оценку $p^*(g)$ распределения вероятностей $p(g)$ случайной величины α, характеристическая функция которой $\widehat{p}^*(\chi)$ соответствует

$$\min_{\widehat{p}(\chi)}\sum_{\chi\neq 1}\Big(\sum_j\left|\widehat{p}(\chi)\widehat{\pi}_j^*(\chi)-\widehat{P}_j^*(\chi)\right|^2+\gamma^2|\widehat{p}(\chi)|^2\Big),$$

где $\gamma^2>0$. Точность оценки $p^*(g)$ также будем характеризовать средним значением величины δ, определяемой соотношением (2).

Теорема 2. *Если* n_j, $N_j\to\infty$, $n_j=o(N_j)$, $\frac{1}{N_j\gamma^2}=O(1)$, $j=1,\ldots,s$, *то величина* $\mathsf{E}\delta$ *асимптотически эквивалентна значению*, *определяемому формулой* (3).

Далее, рассмотрим две случайные величины $\alpha_-=\alpha_1-\alpha_2$ и $\alpha_+=\alpha_1+\alpha_2$, где α_1, α_2 — независимые случайные величины на конечной абелевой группе G с одним и тем же распределением вероятностей $p(g)$, $g\in G$, которое выбирается случайно и имеет равномерное распределение на множестве распределений вероятностей на группе G, удовлетворяющих условию

$$\sum_g\left(p(g)-|G|^{-1}\right)^2=\rho, \tag{4}$$

где для постоянной величины ρ выполняется неравенство

$$\rho\leqslant|G|^{-1}(|G|-1)^{-1}.$$

Обозначим через $P_-(g)$ и $P_+(g)$, $g \in G$, распределения вероятностей случайных величин α_- и α_+, соответственно. Рассмотрим величины

$$R_- = \sum_g \left(P_-(g) - |G|^{-1}\right)^2, \tag{5}$$

$$R_+ = \sum_g \left(P_+(g) - |G|^{-1}\right)^2. \tag{6}$$

При случайном выборе распределения вероятностей $p(g)$, $g \in G$, случайных величин α_1 и α_2 согласно условию (4) величины R_- и R_+ являются случайными.

Теорема 3. *Случайные величины R_- и R_+, значения которых определены равенствами* (5) *и* (6), *имеют одно и то же распределение вероятностей.*

Наряду со случайной величиной R_- рассмотрим случайную величину $\overline{R}_-$, которая определяется следующим равенством

$$\overline{R}_- = R_- \left(\frac{|G|}{2}\rho^2\right)^{-1}, \tag{7}$$

где случайная величина R_- определена формулой (5).

Теорема 4. *Справедливы равенства*

$$\mathsf{E}\left(\overline{R}_-\right)^k = \frac{\mathsf{E}\left(2\sum_{\chi'}(x^2(\chi'))^2 + \sum_{\chi''}(x^2(\chi''))^2\right)^k}{\mathsf{E}(X^2)^{2k}}, \quad k = 1, 2, \ldots,$$

где:

- *в числителе в соответствующей сумме индекс χ' пробегает все отличные от единичного (т. е. $\chi \neq 1$) характеры $\chi \in \widehat{G}$, для которых $\chi^2 = 1$;*
- *индекс χ'' пробегает множество представителей пар характеров (χ, χ^{-1}), для которых $\chi \neq \chi^{-1}$, $\chi \in \widehat{G}$;*
- *случайные величины $x^2(\chi')$, $x^2(\chi'')$ независимы при различных значениях χ', χ'';*
- *случайные величины $x^2(\chi')$, $x^2(\chi'')$, X^2 имеют распределение «хи-квадрат», соответственно, с одной, двумя и $|G|-1$ степенями свободы.*

Далее, рассмотрим случайную величину α, заданную на конечной абелевой группе G, $|G| = d$, и имеющую распределение вероятностей вида

$$\Pr\{\alpha = g\} = p(g), \quad \sum_{g \in G} p(g) = 1, \quad p(g) \geqslant 0. \tag{8}$$

Пусть $\overline{\alpha}_1 = (\alpha_{11}, \alpha_{12}, \ldots, \alpha_{1n})$ есть реализация n независимых испытаний над α, где $n = ds$, s — некоторое целое натуральное число. Пусть $G = \{g_0, g_1, \ldots, g_{d-1}\}$, где g_0 является нулем группы G, а остальные элементы расположены в некотором фиксированном порядке. Пусть последовательность $\overline{\beta} = (\beta_1, \beta_2, \ldots, \beta_n)$ есть фиксированная последовательность из элементов группы G спецификации $(s, s, \ldots, s)$. (Здесь используется понятие спецификации как аналог понятия первичной спецификации из работы [7], стр. 225). Пусть последовательность $\overline{\beta}_1$ получена из последовательности $\overline{\beta}$ путем перестановки ее элементов. Обозначим эту перестановку через π_1, предполагаем, что перестановка π_1 получена в результате случайного равновероятного выбора из соответствующей симметрической группы. Пусть последовательность $\overline{\sigma} = (\sigma_1, \sigma_2, \ldots, \sigma_n)$ получена из последовательностей α_1 и β_1 путем позначного сложения их элементов как элементов группы G, т. е. $\sigma_i = \alpha_{1i} + \beta_{1i}$, $i = 1, 2, \ldots, n$. Ниже приведена несмещенная оценка по последовательности $\overline{\sigma}$ величины $\rho = \sum_{g \in G} p^2(g)$, предполагая, что распределение вероятностей (8) случайной величины α является неизвестным. С этой целью рассмотрим величину

$$\chi^2 = \frac{1}{s}\sum_{g \in G}(\nu(g) - s)^2 = \sum_{g \in G}\eta^2(g), \tag{9}$$

где $\eta(g) = (\nu(g) - s)/\sqrt{s}$, $g \in G$; $\nu(g)$ — g-показатель спецификации последовательности $\overline{\sigma}$.

Теорема 5. *Случайная величина $1 - \chi^2/d$ является несмещенной оценкой величины ρ распределения вероятностей* (8), *где χ^2 определяется формулой* (9).

В представленных тезисах доклада изложены некоторые результаты работ [3–6].

Список литературы

[1] *Воробьев Н. Н.* Сложение двух независимых случайных величин на конечной абелевой группе. Математический сборник, т. 34(76), вып. 1, 1954, с. 89–126.

[2] *Кириллов А. А.* Элементы теории представлений. М.: Наука, 1978, 343 с.

[3] *Лапшин А. В.* Оценка неравновероятности распределения случайной величины на конечной абелевой группе по сумме ее реализаций с элементами случайной N-перестановки. Пятая Всероссийская школа-коллоквиум по стохастическим методам. Тезисы докладов. Обозрение прикладной и промышленной математики, т. 5, вып. 2, 1998, с. 245.

[4] *Лапшин А. В.* Статистическое оценивание распределения слагаемого по серии наблюдений суммы независимых случайных величин на конечной

абелевой группе. Труды по дискретной математике. М.: Физико-математическая литература, т. 4, 2001, с. 129–148.

[5] *Лапшин А. В.* Об одной характеристике распределения разности двух случайных величин на конечной абелевой группе. Труды по дискретной математике. М.: Физико-математическая литература, т. 7, 2003, с. 114–125.

[6] *Лапшин А. В.* Оценка одного параметра распределения случайной величины на конечной абелевой группе по сумме ее реализаций с элементами случайной перестановки. Труды по дискретной математике. М.: Физико-математическая литература, т. 8, 2004 (принято к опубликованию).

[7] *Сачков В. Н.* Комбинаторные методы дискретной математики. М.: Наука, 1977, 319 с.

[8] *Шерстнев В. И.* Случайная величина, равномерно распределенная на конечной абелевой группе, как сумма независимых слагаемых. Теория вероятностей и ее применения, т. 43, вып. 2, 1998, с. 397–403.

Многогранники в конечной абелевой группе и их криптографические приложения

О. А. Логачёв, А. А. Сальников, В. В. Ященко

1. Многогранники в конечной абелевой группе

Пусть G — конечная абелева группа порядка $\#G = N$ с мультипликативной записью операции, $\widehat{G}$ — группа ее характеров. Для каждого множества $D \subseteq G$ определим комплекснозначную функцию Z_D на $\widehat{G}$ с помощью равенства

$$Z_D(\chi) = \sum_{x \in D} \chi(x).$$

По значениям этой функции однозначно восстанавливается множество D, так как в силу соотношения ортогональности для характеров [3]:

$$\frac{1}{N} \sum_{\chi \in \widehat{G}} Z_D(\chi)\chi(y^{-1}) = \begin{cases} 1, & \text{если } y \in D, \\ 0, & \text{если } y \notin D. \end{cases}$$

Для чисел $Z_D(\chi)$ выполняется аналог равенства Парсеваля для коэффициентов Фурье функций на конечной абелевой группе [3]:

$$\sum_{\chi \in \widehat{G}} |Z_D(\chi)|^2 = \sum_{\chi \in \widehat{G}} \sum_{x,y \in D} \chi(x)\overline{\chi}(y) = \sum_{x,y \in D} \Big(\sum_{\chi \in \widehat{G}} \chi\big(xy^{-1}\big) \Big) = N \cdot \#D. \quad (1)$$

Равенство (1) позволяет доказать следующее полезное утверждение.

Лемма 1. *Множество D совпадает с группой G тогда и только тогда, когда*

$$Z_D(\chi) = \sum_{x \in D} \chi(x) = 0 \quad \textit{для любого } \chi \in \widehat{G},\ \chi \neq \chi_0.$$

Работа поддержана Российским фондом фундаментальных исследований (номера проектов 02-01-00581 и 02-01-00687).

Доказательство. Выделив в сумме $\sum_{\chi\in\widehat{G}} |Z_D(\chi)|^2$ слагаемое для $\chi=\chi_0$, из (1) получим

$$\sum_{\substack{\chi\in\widehat{G}\\ \chi\neq\chi_0}} |Z_D(\chi)|^2 = \#D(N-\#D). \tag{2}$$

Значит, $\#D=N$ тогда и только тогда, когда

$$Z_D(\chi)=0 \quad \text{для любого } \chi\in\widehat{G},\ \ \chi\neq\chi_0. \quad \square$$

Среди всех множеств элементов конечной абелевой группы выделяются смежные классы по ее подгруппам, которые мы для удобства будем называть *многогранниками*. Это название позволяет в некоторых случаях опираться на интуитивные геометрические аналогии. Полезным примером при изучении строения множеств элементов конечной абелевой группы является их «локализация» с помощью погружения в многогранник наименьшего размера. Формально задача «локализации» формулируется так: для данного множества $D\subseteq G$ найти минимальную подгруппу $H<G$ такую, что $D\subseteq x_0H$ при некотором $x_0\in D$. Для решения этой задачи введем следующие обозначения:

$$\widehat{D}=\{\chi\in\widehat{G}\mid \chi(x)=\text{const для любого } x\in D\},$$
$$A(\widehat{D})=\{x\in G\mid \chi(x)=1 \text{ для любого } \chi\in\widehat{D}\}.$$

Очевидно, что $\widehat{D}$ — подгруппа $\widehat{G}$, а $A(\widehat{D})$ — подгруппа G.

Теорема 1. *$A(\widehat{D})$ — минимальная из подгрупп H группы G, для которых выполнено включение*

$$x_0^{-1}D\subseteq H \quad \textit{при некотором } x_0\in D.$$

Доказательство. Каждой подгруппе H группы G сопоставим подгруппу H^* группы $\widehat{G}$

$$H^*=\{\chi\in\widehat{G}\mid \chi(x)=1 \text{ для любого } x\in H\}.$$

Известно [1], что отображение $H\mapsto H^*$ является антиизоморфизмом решеток подгрупп группы G и группы $\widehat{G}$. Для данного множества $D\subseteq G$ рассмотрим два множества подгрупп

$$\mathfrak{G}(D)=\{H<G\mid x_0^{-1}D\subseteq H \text{ при некотором } x_0\in D\},$$
$$\widehat{\mathfrak{G}}(D)=\text{множество всех подгрупп группы } \widehat{D}.$$

Покажем, что отображение $H\mapsto H^*$ является антиизоморфизмом $\mathfrak{G}(D)$ и $\widehat{\mathfrak{G}}(D)$. В самом деле, если $H\in\mathfrak{G}(D)$, то для любого $\chi\in H^*$ и любого $y\in D$ выполнено равенство $\chi(x_0^{-1}y)=1$, эквивалентное тому, что

любой характер $\chi\in H^*$ постоянен на множестве D, а это и значит, что $H^*\in\widehat{\mathfrak{G}}(D)$. Легко проверить и обратное включение. Заметим теперь, что

$$\big(A(\widehat{D})\big)^*=\widehat{D}.$$

Поскольку отображение $H\mapsto H^*$ — антиизоморфизм, а $\widehat{D}$ — максимальный элемент решетки $\widehat{\mathfrak{G}}(D)$, то $A(\widehat{D})$ — минимальный элемент решетки $\mathfrak{G}(D)$. $\square$

Заметим теперь, что множество $\widehat{D}$ удобно искать с помощью функции $Z_D(\chi)$, пользуясь следующим очевидным соотношением

$$\widehat{D}=\{\chi\in\widehat{G}\mid |Z_D(\chi)|=\#D\}.$$

Кроме того, модуль функции $Z_D(\chi)$ постоянен на смежных классах $\widehat{G}$ по $\widehat{D}$, т. к. для любого $\chi'\in\widehat{D}$, любого $\chi\in\widehat{G}$ и некоторого $x_0\in D$ выполнено

$$Z_D(\chi\chi')=Z_D(\chi)\chi'(x_0).$$

Очевидно, в последнем равенстве достаточно считать, что χ пробегает множество характеров группы $A(\widehat{D})$. Пользуясь этими замечаниями, леммой 1 и теоремой 1, легко доказать следующую теорему.

Теорема 2. *Множество $D\subseteq G$ является многогранником в группе G тогда и только тогда, когда модуль функции $Z_D(\chi)$ принимает на $\widehat{G}$ только два значения*: 0 *и* $\#D$.

2. Приложения к булевым функциям

Рассмотрим теперь приложения полученных результатов к исследованию криптографических свойств булевых функций. Будем использовать обозначения и результаты работы [2]. Пусть $G=V_n$ — векторное пространство размерности n над полем из двух элементов,

$$\widehat{G}=\{(-1)^{\langle\boldsymbol{\alpha},\boldsymbol{x}\rangle}\mid \boldsymbol{\alpha},\boldsymbol{x}\in V_n\},\quad \langle\boldsymbol{\alpha},\boldsymbol{x}\rangle=\alpha_1x_1+\ldots+\alpha_nx_n,$$

$D=\{\boldsymbol{x}\in V_n\mid f(\boldsymbol{x})=1\}$, где $f(\boldsymbol{x})$ — булева функция. Тогда

$$Z_D(\chi)=Z_f(\boldsymbol{\alpha})=\sum_{\boldsymbol{x}:f(\boldsymbol{x})=1}(-1)^{\langle\boldsymbol{\alpha},\boldsymbol{x}\rangle}=$$
$$=\frac{1}{2}\Big(\sum_{\boldsymbol{x}\in V_n}(-1)^{\langle\boldsymbol{\alpha},\boldsymbol{x}\rangle}-\sum_{\boldsymbol{x}\in V_n}(-1)^{f(\boldsymbol{x})+\langle\boldsymbol{\alpha},\boldsymbol{x}\rangle}\Big)=$$
$$=\begin{cases}2^{n-1}-\frac{1}{2}W_f(\boldsymbol{0}), & \boldsymbol{\alpha}=\boldsymbol{0};\\ -\frac{1}{2}W_f(\boldsymbol{\alpha}), & \boldsymbol{\alpha}\neq\boldsymbol{0},\end{cases}$$

где $W_f(\boldsymbol{\alpha})$ — коэффициенты Уолша — Адамара, и поэтому

$$\widehat{D} = \{\boldsymbol{\alpha} \in V_n \mid \boldsymbol{\alpha} \neq \mathbf{0},\ \ |W_f(\boldsymbol{\alpha})| = 2^n - W_f(\mathbf{0})\} \cup \{\mathbf{0}\}. \tag{3}$$

Многогранники в V_n — это смежные классы по подпространствам V_n или, на языке конечных геометрий, плоскости.

Утверждение теоремы 2 для булева случая выглядит следующим образом.

Следствие 1. *Функция $f(\boldsymbol{x})$ является функцией-индикатором смежного класса по подпространству размерности k тогда и только тогда, когда $|W_f(\boldsymbol{\alpha})|$ при всех $\boldsymbol{\alpha} \in V_n$, $\boldsymbol{\alpha} \neq \mathbf{0}$ принимает только два значения* 0 *и* 2^{k+1}.

Пусть $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_k \in V_n$ — некоторый базис подпространства $\widehat{D}$, заданного условием (3), $\boldsymbol{x}_0 \in V_n$ — произвольный вектор с условием $f(\boldsymbol{x}_0) = 1$, тогда

$$A\big(\widehat{D}\big) = \{\boldsymbol{x} \in V_n \mid \langle \boldsymbol{\alpha}_i, \boldsymbol{x}\rangle = 0,\ \ i = 1, \ldots, k\},$$

а теорема 1 утверждает, что если $f(\boldsymbol{x}) = 1$, то $\langle \boldsymbol{\alpha}_i, \boldsymbol{x}\rangle = \langle \boldsymbol{\alpha}_i, \boldsymbol{x}_0\rangle$, $i = 1, \ldots, k$. Другими словами, множество единиц булевой функции $f(\boldsymbol{x})$ сосредоточено внутри плоскости, заданной линейными уравнениями

$$\begin{cases} \langle \boldsymbol{\alpha}_1, \boldsymbol{x}\rangle = \langle \boldsymbol{\alpha}_1, \boldsymbol{x}_0\rangle, \\ \quad \ldots\ldots\ldots\ldots\ldots \\ \langle \boldsymbol{\alpha}_k, \boldsymbol{x}\rangle = \langle \boldsymbol{\alpha}_k, \boldsymbol{x}_0\rangle. \end{cases} \tag{4}$$

Таким образом, для случая булевых функций теореме 1 можно придать следующую эквивалентную функциональную формулировку.

Теорема 3. *Для любой булевой функции $f(\boldsymbol{x})$ существует декомпозиция вида*

$$f(x_1, \ldots, x_n) = (\langle \boldsymbol{\alpha}_1, \boldsymbol{x} + \boldsymbol{x}_0\rangle + 1)\ \ldots (\langle \boldsymbol{\alpha}_k, \boldsymbol{x} + \boldsymbol{x}_0\rangle + 1) \times \\ \times\, \psi(\langle \boldsymbol{\alpha}_{k+1}, \boldsymbol{x}\rangle, \ldots, \langle \boldsymbol{\alpha}_n, \boldsymbol{x}\rangle). \tag{5}$$

Декомпозиция (5) *однозначно определяется по функции $f(\boldsymbol{x})$ в следующем смысле*

- *$\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_k$ — произвольный базис некоторого подпространства, однозначно определяемого по функции $f(\boldsymbol{x})$;*
- *$\boldsymbol{\alpha}_{k+1}, \ldots, \boldsymbol{\alpha}_n$ — произвольное дополнение $\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_k$ до базиса V_n;*
- *$\boldsymbol{x}_0 \in V_n$ — произвольный вектор с условием $f(\boldsymbol{x}) = 1$;*
- *булева функция $\psi(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n-k})$ от $n - k$ переменных уже не допускает дальнейшей декомпозиции вида* (5) *и однозначно определяется по функции $f(\boldsymbol{x})$.*

Естественно, что после нахождения представления (5) дальнейшее исследование некоторых свойств функции $f(\boldsymbol{x})$ упрощается. В частности, декомпозиции (5) связана с исследованием группы аннуляторов булевой функции, которая введена в работах [6–7] для развития алгебраического метода криптографического анализа. Функция $\varphi(\boldsymbol{x})$ называется *аннулятором* функции $f(\boldsymbol{x})$, если для всех $\boldsymbol{x} \in V_n$ выполнено равенство

$$\varphi(\boldsymbol{x}) f(\boldsymbol{x}) = 0.$$

Из предыдущих рассуждений очевидно следующее утверждение.

Следствие 2. *Пусть для функции $f(\boldsymbol{x})$ найдена декомпозиция* (5). *Тогда группа аффинных аннуляторов функции $f(\boldsymbol{x})$ совпадает с линейной оболочкой множества аффинных функций $\langle \boldsymbol{\alpha}_1, \boldsymbol{x}\rangle + \langle \boldsymbol{\alpha}_1, \boldsymbol{x}_0\rangle, \ldots, \langle \boldsymbol{\alpha}_k, \boldsymbol{x}\rangle + \langle \boldsymbol{\alpha}_k, \boldsymbol{x}_0\rangle$, а функция $\psi(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_{n-k})$ имеет тривиальную группу аффинных аннуляторов* (*т. е. только нулевую функцию*).

Таким образом, предыдущие построения дают алгоритм поиска всех аффинных аннуляторов данной булевой функции $f(\boldsymbol{x})$, если известны ее коэффициенты Уолша — Адамара $W_f(\boldsymbol{\alpha})$. Целесообразно найти описание и других аннуляторов функции $f(\boldsymbol{x})$ на языке коэффициентов Уолша — Адамара.

Теорема 4. *Булева функция $\varphi(\boldsymbol{x})$ является аннулятором булевой функции $f(\boldsymbol{x})$ тогда и только тогда, когда выполнено одно из двух эквивалентных условий*:

1) $W_f(\mathbf{0}) + W_\varphi(\mathbf{0}) - W_{f+\varphi}(\mathbf{0}) = 2^n$;
2) $W_f(\boldsymbol{\alpha}) + W_\varphi(\boldsymbol{\alpha}) - W_{f+\varphi}(\boldsymbol{\alpha}) = 0$ *для любого* $\boldsymbol{\alpha} \in V_n$, $\boldsymbol{\alpha} \neq \mathbf{0}$.

Доказательство. Пользуясь очевидным соотношением

$$(-1)^{f(\boldsymbol{x})\varphi(\boldsymbol{x})} = \frac{(-1)^0 + (-1)^{f(\boldsymbol{x})} + (-1)^{\varphi(\boldsymbol{x})} - (-1)^{f(\boldsymbol{x})+\varphi(\boldsymbol{x})}}{2},$$

получаем для любого $\boldsymbol{\alpha} \in V_n$

$$W_{f\varphi}(\boldsymbol{\alpha}) = \frac{1}{2}\big(W_0(\boldsymbol{\alpha}) + W_f(\boldsymbol{\alpha}) + W_\varphi(\boldsymbol{\alpha}) - W_{f+\varphi}(\boldsymbol{\alpha})\big). \tag{6}$$

Поскольку $f(\boldsymbol{x})\varphi(\boldsymbol{x}) \equiv 0$ тогда и только тогда, когда выполнено одно из двух эквивалентных условий:

1) $W_{f\varphi}(\mathbf{0}) = 2^n$;
2) $W_{f\varphi}(\mathbf{0}) = 0$ для любого $\boldsymbol{\alpha} \in V_n$, $\boldsymbol{\alpha} \neq \mathbf{0}$,

из равенства (6) получаем утверждение теоремы. □

Иногда удобно записывать условия теоремы 4 в следующем более симметрическом виде

1) $W_{f+\varphi+1}(\mathbf{0}) = 2^n - (W_f(\mathbf{0}) + W_\varphi(\mathbf{0}))$;
2) $W_{f+\varphi+1}(\boldsymbol{\alpha}) = -(W_f(\boldsymbol{\alpha}) + W_\varphi(\boldsymbol{\alpha}))$ для любого $\boldsymbol{\alpha} \in V_n$, $\boldsymbol{\alpha} \neq \mathbf{0}$.

Легко видеть, что если $f(\boldsymbol{x})\varphi(\boldsymbol{x}) \equiv 0$, то $W_f(\mathbf{0}) + W_\varphi(\mathbf{0}) \geqslant 0$, причем $W_f(\mathbf{0}) + W_\varphi(\mathbf{0}) = 0$ тогда и только тогда, когда $f(\boldsymbol{x}) + \varphi(\boldsymbol{x}) \equiv 1$. Аннулятор $f(\boldsymbol{x}) + 1$ будем называть *тривиальным*.

С помощью простых преобразований из условий теоремы 4 легко получить

Следствие 3. *Функция*

$$\varphi(\boldsymbol{x}) = \begin{cases} 1, & \text{если } \boldsymbol{x} \in L + \boldsymbol{x}_0; \\ 0, & \text{если } \boldsymbol{x} \notin L + \boldsymbol{x}_0, \end{cases}$$

является аннулятором булевой функции $f(\boldsymbol{x})$ тогда и только тогда, когда

$$\sum_{\boldsymbol{\gamma} \in L^\perp} (-1)^{\langle \boldsymbol{\gamma},\, \boldsymbol{x}_0 \rangle} W_f(\boldsymbol{\gamma}) = 2^n.$$

Следствие 4. *Неаффинные платовидные и неаффинные уравновешенные булевы функции не имеют аффинных аннуляторов.*

Доказательство. Все аффинные аннуляторы $\langle \boldsymbol{\alpha}, \boldsymbol{x} \rangle + \varepsilon$, $\boldsymbol{\alpha} \neq \mathbf{0}$ функции $f(\boldsymbol{x})$ описываются условиями (3):

$$|W_f(\boldsymbol{\alpha})| = 2^n - W_f(\mathbf{0}).$$

Если функция f уравновешена, т. е. $W_f(\mathbf{0}) = 0$, и имеет аффинный аннулятор, то $|W_f(\boldsymbol{\alpha})| = 2^n$, т. е. f — аффинная.

Платовидные функции порядка $2r$ определяются условием [8], [4]:

$$|W_f(\boldsymbol{\alpha})| \text{ принимает только два значения: } 0,\ \ 2^{n-r}.$$

Тогда, очевидно, что если $r \neq 0$, т. е. функция неаффинная, то условие (3) не выполняется ни при каком $\boldsymbol{\alpha}$. □

В частности, квадратичные булевы функции не имеют аффинных аннуляторов.

Следствие 5. *Две неаффинные платовидные функции с непересекающимися носителями не аннулируют друг друга.*

Доказательство. Поскольку носители f и φ не пересекаются, т. е. $W_f(\boldsymbol{\alpha}) W_\varphi(\boldsymbol{\alpha}) = 0$ для любого $\boldsymbol{\alpha} \in V_n$, то

$$W_{f+\varphi}(\mathbf{0}) = \frac{1}{2^n} \sum_{\boldsymbol{\alpha} \in V_n} W_f(\boldsymbol{\alpha}) W_\varphi(\boldsymbol{\alpha}) = 0.$$

Поэтому f и φ аннулируют друг друга тогда и только тогда, когда

$$W_f(\mathbf{0}) + W_\varphi(\mathbf{0}) = 2^n,$$

что невозможно в силу условий следствия 5. □

Лемма 2. *Пусть $f(\boldsymbol{x})\varphi(\boldsymbol{x}) \equiv 0$ и $f(\boldsymbol{x})$, $\varphi(\boldsymbol{x})$ — бент-функции, $f(\boldsymbol{x}) + \varphi(\boldsymbol{x}) + 1 \not\equiv 0$. Тогда $f(\boldsymbol{x}) + \varphi(\boldsymbol{x}) + 1$ — функция-индикатор смежного класса по подпространству размерности $n/2$.*

Доказательство. В силу условий теоремы 4 и условий леммы, очевидно, имеем

$$W_{f+\varphi+1}(\mathbf{0}) = 2^n - 2^{n/2+1},$$
$$W_{f+\varphi+1}(\boldsymbol{\alpha}) \in \left\{0,\, \pm 2^{n/2+1}\right\} \text{ при любом } \boldsymbol{\alpha} \neq \mathbf{0}.$$

Отсюда, с учетом следствия 1 и вытекает лемма. □

Следствие 6. *Если $f(\boldsymbol{x})$, $\varphi(\boldsymbol{x})$ — бент-функции степени нелинейности меньшей $n/2$ и $f(\boldsymbol{x}) + \varphi(\boldsymbol{x}) \not\equiv 1$, то функции $f(\boldsymbol{x})$ и $\varphi(\boldsymbol{x})$ не аннулируют друг друга.*

Лемма 3. *Пусть $f(\boldsymbol{x})$ — платовидная функция порядка $2r$, а $\varphi(\boldsymbol{x})$ — платовидная функция порядка $2s$. Если $r \geqslant 2$, $s \geqslant r+1$, или $r = 1$, $s \geqslant 3$, то функции $f(\boldsymbol{x})$ и $\varphi(\boldsymbol{x})$ не аннулируют друг друга.*

Доказательство. Предположим противное. Тогда в силу условий теоремы 4 имеем

$$|2^n - (W_f(\mathbf{0}) + W_\varphi(\mathbf{0}))| = |W_{f+\varphi+1}(\mathbf{0})| \leqslant \max_{\boldsymbol{\alpha} \in V_n} |W_\varphi(\boldsymbol{\alpha})|,$$
$$\frac{1}{2^n} \sum_{\boldsymbol{\alpha} \in V_n} |W_\varphi(\boldsymbol{\alpha})| = 2^{n-s} \cdot \frac{1}{2^n} \cdot 2^{n-r} \cdot 2^{2r} = 2^{n-s+r}. \qquad (7)$$

Очевидно, что для значения суммы $W_f(\mathbf{0}) + W_f(\mathbf{0})$ возможны 4 варианта. Расположим их в порядке возрастания:

$$2^{n-s} \leqslant 2^{n-r} - 2^{n-s} < 2^{n-r} < 2^{n-s} + 2^{n-r}.$$

Поэтому наименьшее возможное значение для $|2^n - (W_f(\mathbf{0}) + W_\varphi(\mathbf{0}))|$ равно $2^n - 2^{n-s} - 2^{n-r}$. Если теперь в условиях леммы мы докажем неравенство

$$2^n - (2^{n-s} + 2^{n-r}) > 2^{n-s+r}, \qquad (8)$$

то получим противоречие с неравенством (7) и тем самым докажем лемму. Неравенство (8) эквивалентно неравенству

$$2^n - 2^{n-s+r} > 2^{n-s} + 2^{n-r},$$

которое эквивалентно неравенству

$$2^r > \frac{2^s + 2^r}{2^s - 2^r} = 1 + \frac{2}{2^{s-r} - 1}.$$

Правая часть последнего неравенства монотонно убывает при возрастании $s - r$ и поэтому, очевидно, в условиях леммы выполнено неравенство (7). □

Замечание. Для ответа на вопрос, когда две платовидные функции могут аннулировать друг друга, необходимо рассмотреть еще два случая

1) $r = 1$, $s = 2$;
2) $r = s$.

Список литературы

[1] *Биркгоф Г.* Теория решеток. М.: Наука, 1984.

[2] *Логачёв О. А.*, *Сальников А. А.*, *Ященко В. В.* Булевы функции в теории кодирования и криптологии. М.: МЦНМО, 2004.

[3] *Серр Ж.-П.* Линейные представления конечных групп. М.: Мир, 1970.

[4] *Carlet C.*, *Prauff I. E.* On Plateaued Functions and their Constructions. Proc. FSE'2003.

[5] *Courtois N. T.* Fast Algebraic Attacks on Stream Cipher with Linear Feedback. Proc. CRYPTO'2003, LNCS, v. 2729, pp. 176–194, 2003.

[6] *Courtois N. T.*, *Meier W.* Algebraic Attacks on Stream Cipher with Linear Feedback. Proc. EUROCRYPT'2003, LNCS, v. 2656, pp. 345–369, 2003.

[7] *Meier W.*, *Pasalic E.*, *Carlet C.* Algebraic Attacks and Decomposition of Boolean Functions. Proc. EUROCRYPT'2004, LNCS, v. 3027, pp. 474–491, 2004.

[8] *Zheng Y.*, *Zhang X.-M.* On Plateaued Functions. IEEE Trans. on Information Theory, v. 47, no. 3, pp. 1215–1223, 2001.

Разрешимость задачи дискретного логарифмирования в кольце вычетов по составному модулю

А. В. Маркелова

В некоторых криптографических протоколах возникает задача дискретного логарифмирования по нескольким несвязанным модулям (например, при реализации электронной монеты, где центр сертификации, банк и клиент имеют ключи по различным модулям). Поэтому на практике может оказаться полезной задача выяснения совместности системы

$$\begin{cases} a^x \equiv b \pmod{p}, \\ a^x \equiv b \pmod{q}, \end{cases}$$

которая равносильна сравнению

$$a^x \equiv b \pmod{pq}.$$

Формулируя задачу в более общем виде, будем рассматривать разрешимость сравнения

$$a^x \equiv b \pmod{M},$$

где $M = p_1^{\alpha_1} \dots p_k^{\alpha_k}$, $a \in (Z/MZ)^*$. Этот вопрос рассматривался ранее О. Н. Василенко в работе [1]. Приведенные ниже результаты применимы в некотором более общем, чем в этой работе, случае.

Лемма 1. *Если* $M \in \{2, 4, p^\alpha, 2p^\alpha\}$ $(p \neq 2)$ *и* $a \in (Z/MZ)^*$, *то сравнение* $a^x \equiv b \pmod{M}$ *разрешимо тогда и только тогда, когда* $b \in (Z/MZ)^*$ *и* $\operatorname{ord}_M b \mid \operatorname{ord}_M a$.

Эта лемма легко следует из цикличности группы $(Z/MZ)^*$ при указанных M.

Напомним, что $Q(a, r) = (a^{\lambda(r)} - 1)/r \pmod{r}$ — частное Ферма, где $\lambda(r)$ — функция Кармайкла (наименьшее общее кратное порядков элементов в группе $(Z/rZ)^*$).

Следующие известные свойства частного Ферма доказаны, например, в [2] (стр. 146, леммы 5.12 и 5.15).

Лемма 2.

1. (*показательное свойство частного Ферма*). *Пусть* $(a, r) = (b, r) = 1$. *Тогда* $Q(ab, r) \equiv Q(a, r) + Q(b, r) \pmod{r}$.

2. *Число* $R = r^2/(\lambda(r),\ r)$ *является периодом* $Q(x,\ r)$.

Лемма 3. *Пусть* $a \in (Z/p^\alpha Z)^*$ $(\alpha \geqslant 1)$. *Тогда* $\operatorname{ord}_p a = (\operatorname{ord}_{p^\alpha} a,\ p-1)$.

Доказательство. Если $p = 2$, то утверждение тривиально.

Пусть p — нечетное. Обозначим $d = (\operatorname{ord}_{p^\alpha} a,\ p-1)$, $\delta = \operatorname{ord}_p a$. Поскольку $a^{\operatorname{ord}_{p^\alpha} a} \equiv 1 \pmod{p^\alpha}$, то

$$a^{\operatorname{ord}_{p^\alpha} a} \equiv 1 \pmod{p}.$$

Следовательно, $\delta \mid \operatorname{ord}_{p^\alpha} a$. Учитывая, что $\delta \mid p-1$, получим

$$\delta \mid (\operatorname{ord}_{p^\alpha} a,\ p-1) = d.$$

Заметим, что $\operatorname{ord}_{p^\alpha} a = p^\beta d$ $(0 \leqslant \beta \leqslant \alpha - 1)$. Так как

$$a^\delta = 1 + p^\gamma q_0 \quad (p \nmid q_0),$$

то, возводя по биному Ньютона в степень p, получим

$$a^{p\delta} = 1 + p^{\gamma+1} q_1 \quad (p \nmid q_1).$$

Далее по индукции

$$a^{p^\beta \delta} = 1 + p^{\gamma+\beta} q_2 \quad (p \nmid q_2).$$

Из того, что $\delta \mid d$ и $\left(\dfrac{d}{\delta},\ p\right) = 1$, следует, что

$$(a^{p^\beta \delta})^{\frac{d}{\delta}} = 1 + p^{\gamma+\beta} q_3 \quad (p \nmid q_3).$$

Но поскольку $a^{p^\beta d} \equiv 1 \pmod{p^\alpha}$, то и

$$a^{p^\beta \delta} \equiv 1 \pmod{p^\alpha},$$

то есть $d \mid \delta$. □

Лемма 4. *Пусть порядок* a *по модулю* p^α $(\alpha > 1)$ *равен* $\delta = p^\beta \delta_1$, *где* $\delta_1 = (\delta,\ p-1)$, $0 \leqslant \beta \leqslant \alpha - 1$. *Тогда если* $\beta = 0$, *то* $Q(a,\ p^{\alpha-1}) \equiv 0 \pmod{p^{\alpha-1}}$, *а если* $\beta \neq 0$, *то* $p^{\alpha-\beta-1} \parallel Q(a,\ p^{\alpha-1})$ (*в точности делит*).

Доказательство. По лемме 3

$$a^{\delta_1} = 1 + p^\gamma q_1,$$

где $q_1 = 0$ или $p \nmid q_1$. Тогда по биному Ньютона получаем, что

$$a^{p^{\alpha-2}(p-1)} = 1 + p^{\gamma+\alpha-2} q_2,$$

где $p \nmid q_2 \iff p \nmid q_1$. Следовательно,

$$Q(a,\ p^{\alpha-1}) = \frac{1 + p^{\gamma+\alpha-2} q_2 - 1}{p^{\alpha-1}} = p^{\gamma-1} q_2.$$

Если $\beta = 0$, то $\gamma \geqslant \alpha$ или $q_1 = 0$, поэтому $Q(a,\ p^{\alpha-1}) \equiv 0 \pmod{p^{\alpha-1}}$. При $\beta \neq 0$ из первого равенства получим, что

$$a^{p^\beta \delta_1} = 1 + p^{\gamma+\beta} q_3,$$

где $p \nmid q_3$. Учитывая, что $\operatorname{ord}_{p^\alpha} a = p^\beta \delta_1$, получаем $\alpha = \gamma + \beta$, то есть, $p^{\alpha-\beta-1} \parallel Q(a,\ p^{\alpha-1})$. □

Теорема 1. *Пусть* p — *простое нечетное число*, $a, b \in (Z/p^\alpha Z)^*$, $\alpha > 1$. *Тогда сравнение* $a^x \equiv b \pmod{p^\alpha}$ *равносильно системе*

$$\begin{cases} Q(a,\ p^{\alpha-1}) x \equiv Q(b,\ p^{\alpha-1}) \pmod{p^{\alpha-1}}, \\ a^x \equiv b \pmod{p}. \end{cases} \tag{1}$$

Доказательство. Поскольку p^α является периодом для $Q(a, p^{\alpha-1})$ (лемма 2), то, с учетом показательного свойства частного Ферма, решение исходного сравнения удовлетворяет системе (1). Значит достаточно доказать, что, во-первых, исходное сравнение и система разрешимы одновременно, и во-вторых, что в случае разрешимости система имеет единственное по модулю порядка a решение.

Итак, пусть $\operatorname{ord}_{p^\alpha} a = p^{n_1} \delta_1$ и $\operatorname{ord}_{p^\alpha} b = p^{n_2} \delta_2$ $((\delta_1,\ p) = (\delta_2,\ p) = 1$; $n_1, n_2 \leqslant \alpha - 1)$. По лемме 1, исходное сравнение разрешимо тогда и только тогда, когда $\operatorname{ord}_{p^\alpha} b \mid \operatorname{ord}_{p^\alpha} a$, то есть $\delta_2 | \delta_1$ и $0 \leqslant n_2 \leqslant n_1$.

По леммам 1 и 3, второе сравнение системы (1) разрешимо, если $\delta_2 \mid \delta_1$, и его решение является единственным по модулю δ_1.

Применяя лемму 4, получаем, что первое сравнение системы разрешимо тогда и только тогда, когда $0 \leqslant n_2 \leqslant n_1$, при этом решение является единственным по модулю p^{n_1}. То есть по КТО получаем, что система (1) имеет единственное решение по модулю $\operatorname{ord}_{p^\alpha} a$. □

Теорема 2. *Пусть* $\alpha \geqslant 5$, $a, b \in (Z/2^\alpha Z)^*$. *Тогда сравнение* $a^x \equiv b \pmod{2^\alpha}$ *равносильно системе*

$$\begin{cases} Q(a,\ 2^{\alpha-2}) x \equiv Q(b,\ 2^{\alpha-2}) \pmod{2^{\alpha-2}}, \\ a^x \equiv b \pmod{4}. \end{cases} \tag{2}$$

Доказательство. Пусть

$$a \equiv (-1)^{\delta_a} 5^{\gamma_a} \pmod{2^\alpha}, \qquad b \equiv (-1)^{\delta_b} 5^{\gamma_b} \pmod{2^\alpha}.$$

Поскольку, по лемме 2,

$$Q(a,\ 2^{\alpha-2}) = \frac{1 - ((-1)^{\delta_a} 5^{\gamma_a})^{2^{\alpha-4}}}{2^{\alpha-2}} = \frac{1 - (5^{\gamma_a})^{2^{\alpha-4}}}{2^{\alpha-2}} =$$
$$= Q(5^{\gamma_a},\ 2^{\alpha-2}) = \gamma_a Q(5,\ 2^{\alpha-2}),$$

то первое сравнение системы (2) перепишется в виде:

$$\gamma_a Q(5,\, 2^{\alpha-2})x \equiv \gamma_b Q(5,\, 2^{\alpha-2}) \pmod{2^{\alpha-2}}.$$

Учитывая, что $Q(5, 2^{\alpha-2}) \equiv \left(5^{2^{\alpha-4}} - 1\right)/(2^{\alpha-2}) \pmod{2^{\alpha-2}}$ — нечетное (см. [2], стр. 148, теорема 5.20) и переписывая второе сравнение, получаем систему

$$\begin{cases} \gamma_a x \equiv \gamma_b \pmod{2^{\alpha-2}}, \\ \delta_a x \equiv \delta_b \pmod 2, \end{cases}$$

равносильную исходному сравнению. □

Пусть теперь нам нужно выяснить разрешимость сравнения

$$a^x \equiv b \pmod{p_1^{\alpha_1} \dots p_k^{\alpha_k}}, \tag{3}$$

где $a,\, b \in (Z/p_1^{\alpha_1} \dots p_k^{\alpha_k} Z)^*$, $2 \leqslant p_1 < \dots < p_k$. Обозначим $\delta_i = p_i^{\beta_i}\delta_i'$ — порядок элемента a по модулю $p_i^{\alpha_i}$, $(p_i,\, \delta_i') = 1$. Определим γ_{ij} так, что $\delta_i = p_j^{\gamma_{ij}}\delta_{ij}$, $(p_j,\, \delta_{ij}) = 1$.

Алгоритм

Дано: $a,\, b \in (Z/p_1^{\alpha_1} \dots p_k^{\alpha_k} Z)^*$.

Требуется: определить разрешимость сравнения (3).

1. Если $p_1 \neq 2$, то $i = 1$, идти на шаг 6.
2. Если $\alpha_1 \geqslant 5$, то идти на шаг 3, иначе найти перебором c_1 — решение сравнения
$$a^x \equiv b \pmod{2^{\alpha_1}}.$$
Если сравнение не разрешимо, то СТОП: сравнение (3) — не разрешимо;
иначе идти на шаг 5.
3. Найти c_1 — наименьшее неотрицательное решение линейного сравнения:
$$Q(a,\, 2^{\alpha_1-2})x \equiv Q(b,\, 2^{\alpha_1-2}) \pmod{2^{\alpha_1-2}}.$$
Если сравнение не разрешимо, то СТОП: сравнение (3) — не разрешимо.
4. Проверить выполнимость сравнения
$$a^{c_1} \equiv b \pmod 4.$$
Если не выполнено, то СТОП: сравнение (3) — не разрешимо.
5. $i = 2$.
6. Проверить выполнение условия:
$$b^{\delta_i'} \equiv 1 \pmod{p_i}.$$
Если не выполняется, то СТОП: сравнение (3) — не разрешимо.

7. Если $\alpha_i = 1$, то $c_i = 1$, иначе найти c_i — наименьшее неотрицательное решение линейного сравнения:
$$Q(a,\, p_i^{\alpha_i-1})x \equiv Q(b,\, p_i^{\alpha_i-1}) \pmod{p_i^{\alpha_i-1}}.$$
Если сравнение неразрешимо, то СТОП: сравнение (3) — не разрешимо.
8. Для каждого $j \in \{1, \dots, i-1\}$ выполнить действия:
   a. Если $p_j \mid \delta_i$, то проверить выполнение сравнения
   $$(ba^{-c_j})^{\delta_{ij} p_j^{\max\{\gamma_{ij} - \beta_j, 0\}}} \equiv 1 \pmod{p_i^{\alpha_i}}.$$
   b. Проверить разрешимость сравнения
   $$a^x \equiv b \pmod{p_i p_j}.$$
   Если не выполняется одно из условий, то СТОП: сравнение (3) — не разрешимо.
9. Если $i < k$, то $i := i + 1$, идти на шаг 6;
иначе СТОП: сравнение (3) — разрешимо.

Теорема 3. *Данный алгоритм проверяет разрешимость сравнения* (3) *за* $O(\log^3 M)$ *арифметических операций и* $O(\log^2 M)$ *проверок разрешимости по модулю* $p_i p_j$.

Доказательство. Сравнение $a^x \equiv b \pmod{p_1^{\alpha_1} \dots p_k^{\alpha_k}}$ равносильно системе:

$$\begin{cases} a^x \equiv b \pmod{p_1^{\alpha_1}}, \\ \dots\dots\dots\dots\dots\dots \\ a^x \equiv b \pmod{p_k^{\alpha_k}}. \end{cases}$$

Разрешимость каждого сравнения системы проверяется на шагах 2, 3, 4 при $p_1 = 2$ и на шагах 6 и 7 для нечетных p_i (по теоремам 1 и 2 и лемме 1). Обозначим x_j — решение сравнения $a^x \equiv b \pmod{p_j^{\alpha_j}}$; $x_j \equiv c_j \pmod{p_j^{\beta_j}}$ (c_j находится шагах 2, 3 или 7 алгоритма по теоремам 1 и 2).

Осталось проверить совместность этой системы. Для этого необходимо и достаточно выполнения $\forall\, i \neq j$ условия

$$x_i \equiv x_j \pmod{(\delta_i,\, \delta_j)}.$$

Поскольку на шаге 8 $i > j$, а следовательно, $p_i > p_j$, то получаем $(\delta_i,\, \delta_j) = \left(p_i^{\beta_i}\delta_i',\, p_j^{\beta_j}\delta_j'\right) = \left(\delta_i,\, p_j^{\beta_j}\right)(\delta_i',\, \delta_j')$, а следовательно, вместо последнего сравнения можно проверять условия

$$\begin{cases} x_i \equiv x_j \pmod{(\delta_i,\, p_j^{\beta_j})}, \\ x_i \equiv x_j \pmod{(\delta_i',\, \delta_j')}. \end{cases}$$

Заметим, что $\left(\delta_i, p_j^{\beta_j}\right) = p_j^{\min(\gamma_{ij},\ \beta_j)}$. Теперь, если $\gamma_{ij} \leqslant \beta_j$, то $\left(\delta_i, p_j^{\beta_j}\right) = p_j^{\gamma_{ij}}$, и для решения необходимо и достаточно, чтобы $x_i \equiv x_j \pmod{p_j^{\gamma_{ij}}}$. Поскольку $x_j \equiv c_j \pmod{p_j^{\gamma_{ij}}}$, то необходимо и достаточно проверить условие $a^{\delta_{ij} c_j} \equiv b^{\delta_{ij}} \pmod{p_i^{\alpha_i}}$, что и проверяется на шаге 8a.

Если же $\gamma_{ij} > \beta_j$, то необходимо и достаточно, чтобы $x_i \equiv x_j \pmod{p_j^{\beta_j}}$, то есть должно существовать такое целое число u, что $x_i \equiv c_j + u p_j^{\beta_j} \pmod{p_j^{\gamma_{ij}}}$. А это так тогда и только тогда, когда следующее сравнение разрешимо относительно $u \in Z$

$$a^{\delta_{ij}\left(c_j + u p_j^{\beta_j}\right)} \equiv b^{\delta_{ij}} \pmod{p_i^{\alpha_i}},$$
$$\left(a^{\delta_{ij} p_j^{\beta_j}}\right)^u \equiv b^{\delta_{ij}} a^{-\delta_{ij} c_j} \pmod{p_i^{\alpha_i}}.$$

Поскольку $p_i \geqslant 3$ $(p_i > p_j \geqslant 2)$ и $\operatorname{ord}\left(a^{\delta_{ij} p_j^{\beta_j}}\right) = p_j^{\gamma_{ij} - \beta_j}$, то по лемме 1 разрешимость этого сравнения эквивалентна тому, что

$$\left(b^{\delta_{ij}} a^{-\delta_{ij} c_j}\right)^{p_j^{\gamma_{ij} - \beta_j}} \equiv 1 \pmod{p_i^{\alpha_i}}$$

(проверяется на шаге 8a). А выполнимость второго условия разрешимости, т. е.

$$x_i \equiv x_j \pmod{(\delta_i', \delta_j')},$$

проверяется условием 8b.

Сложность этого алгоритма — $O(\log^3 M)$ арифметических операций (действительно, два цикла — по i и по j — дают $O(\log^2 M)$, и в каждом цикле наиболее трудоемкой является операция возведения в степень — еще $O(\log M)$ операций; при этом нахождение в начале всех δ_i так же занимает $O(\log^3 M)$ операций) и $O(\log^2 M)$ проверок разрешимости по модулю $p_i p_j$. □

Итак, вопрос о разрешимости исходного сравнения свелся к вопросу о разрешимости сравнения

$$a^x \equiv b \pmod{pq}, \tag{4}$$

где p и q — различные простые, нечетные числа (если одно из них четное, то см. лемму 1). Сформулируем простую лемму.

Лемма 5. *Если* $(\operatorname{ord}_p a, \operatorname{ord}_q a) = 1$, *то сравнение* $a^x \equiv b \pmod{pq}$ *разрешимо тогда и только тогда, когда разрешимы соответствующие сравнения по модулям* p *и* q.

Рассмотрим более общий случай.

Теорема 4. *Пусть* $p = 2r + 1$, $q = 2s + 1$, *где* $(r, s) = 1$, r, s — *нечетные. Тогда* (4) *разрешимо тогда и только тогда, когда*:

I. $a^x \equiv b$ — *разрешимо по модулям* p *и* q.

II. *Выполнено одно из следующих условий*:

1) $\operatorname{ord}_p a \mid r$;

2) $\operatorname{ord}_q a \mid s$;

3) $\left(\frac{b}{pq}\right) = 1$.

Доказательство. Понятно, что если $a^x \equiv b \pmod{pq}$ разрешимо, то I выполнено. Осталось доказать, что при условии выполнения I, исходное сравнение разрешимо тогда и только тогда, когда выполнено II.

1), 2) Поскольку $(\operatorname{ord}_p a, \operatorname{ord}_q a) = 1$, то теорема следует из предыдущей леммы.

3) Обозначим за c_1 и c_2 — решения сравнений $a^x \equiv b \pmod{p}$ и $a^x \equiv b \pmod{q}$. Они определены по модулям $\operatorname{ord}_p a$ и $\operatorname{ord}_q a$ соответственно. Заметим, что если не выполнены условия 1, 2), то

$$(\operatorname{ord}_p a, \operatorname{ord}_q a) = 2.$$

$a^x \equiv b \pmod{pq}$ разрешимо $\Longleftrightarrow c_1 \equiv c_2 \pmod{(\operatorname{ord}_p a, \operatorname{ord}_q a)}$.

Поскольку $2 \mid \operatorname{ord}_p a$ и $2 \mid \operatorname{ord}_q a$, то a — квадратичный невычет по обоим модулям p и q. Нам нужно, чтобы $c_1 \equiv c_2 \pmod 2$. Если c_1 и c_2 — четные, то b — квадратичный вычет по обоим модулям, если оба нечетные, то, соответственно, невычет. То есть условие совместности выполнено тогда и только тогда, когда $\left(\frac{b}{p}\right) = \left(\frac{b}{q}\right)$, то есть $\left(\frac{b}{pq}\right) = 1$. □

Список литературы

[1] *Василенко О. Н.* О разрешимости задачи дискретного логарифмирования в кольцах вычетов // Фундаментальная и прикладная математика, 2002, том 8, № 3, с. 647–653.

[2] *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии — М.: МЦНМО, 2003.

YAQS — еще один алгоритм квадратичного решета

А. Ю. Нестеренко

1. Введение

Решение задачи факторизации составного, свободного от квадратов числа $N \in \mathbb{Z}$, не являющегося степенью простого, основывается на простой идее, авторство которой приписывается П. Ферма. Пусть известна пара целых чисел x и y таких, что

$$u^2 \equiv v^2 \pmod N, \qquad u \not\equiv \pm v \pmod N. \tag{1}$$

Тогда либо $\gcd(u-v, N) > 1$, либо $\gcd(u+v, N) > 1$, что и дает нам искомое решение.

В 1971 году Д. Шенкс (см. статью [10]) для нахождения пары (1) предложил алгоритм, использующий разложение $\sqrt{N}$ в непрерывную дробь. В 1975 году в статье [5] М. Моррисон и Дж. Бриллхарт, основываясь на идеях М. Крайчика [4], предложили модификацию алгоритма Шенкса, позволившую разложить на ЭВМ седьмое число Ферма F_7. В алгоритме Моррисона и Бриллхарта использовалось разложение $\sqrt{N}$ в непрерывную дробь для получения соотношений вида

$$\beta_n \equiv \gamma_n^2 \pmod N, \quad n = 1, 2, \ldots, \tag{2}$$

где для чисел β_n была верна оценка сверху $\beta_n < 2\sqrt{N}$ и известно разложение в произведение маленьких простых чисел

$$\beta_n = \prod_i p_i^{\varepsilon_{i,n}}, \quad \varepsilon_{i,n} \in \mathbb{Z}, \quad i = 1, 2, \ldots. \tag{3}$$

В дальнейшем мы будем называть множество простых чисел, участвующих в разложении (3), факторной базой и обозначать символом $\mathcal{B}$.

Из соотношений (2) впоследствии выбирались такие пары (β_j, γ_j), где $j \in \{1, 2, \ldots\}$, что

$$u^2 = \prod_j \beta_j \equiv \Big(\prod_j \gamma_j\Big)^2 = v^2 \pmod N.$$

В качестве развития алгоритма Моррисона и Бриллхарта предлагалось (см. книгу [2]) выбирать некоторый целочисленный множитель k и раскладывать $\sqrt{kN}$ в непрерывную дробь. Выбор подходящего значения k позволял обеспечить большую вероятность появления соотношений (2) для которых выполнено равенство (3).

В 1981 году К. Померанс (см. статью [9]) для генерации соотношений (2) предложил алгоритм квадратичного решета (QS). Померанс рассматривал целочисленный многочлен $f(x) = (x+s)^2 - N$, где $s = \lceil \sqrt{N} \rceil$, для которого в каждой точке $x \in \mathbb{Z}$ выполнено сравнение

$$\beta = f(x) \equiv \gamma^2 \pmod N, \quad \text{где } \gamma = x + s.$$

Предложенный Померансом метод (квадратичное решето) поиска значений многочлена $f(x)$, для которых известно разложение (3), позволил существенно снизить трудоемкость алгоритма по сравнению с алгоритмом Моррисона и Бриллхарта.

Коротко, алгоритм квадратичного решета, применительно к произвольному многочлену $f(x)$, заключается в следующем. Если для простого числа p выполнено равенство (3), то из условия $p \mid \beta$ следует, что найдется такое целое значение x_p, что $\beta = f(x_p) \equiv 0 \pmod p$. Тогда каждому простому числу p из факторной базы $\mathcal{B}$ соответствует множество целых чисел $\{x_p + np\}$, где $n \in \mathbb{Z}$ пробегает некоторый заранее заданный интервал, таких, что $f(x_p + np) \equiv 0 \pmod p$. Рассмотрев пересечение этих множеств для всех простых чисел p из $\mathcal{B}$, мы получим целые точки в которых значения многочлена $\beta = f(x)$, с большой вероятностью, удовлетворяют равенству (3).

В 1987 году Д. Сильверман (см. статью [11]) предложил модификацию алгоритма Померанса (MPQS), основанную на применении квадратичного решета к нескольким многочленам для получения соотношений (2). Воспользовавшись идеей П. Монтгомери (см. [9], а также [1]), Сильверман предложил рассматривать такие многочлены $f(x) \in \mathbb{Z}[x]$, $f(x) = ax^2 + bx + c$, что

$$D_f = b^2 - 4ac \equiv 0 \pmod N.$$

Тогда выполнено сравнение $\beta = af(x) \equiv (ax+b)^2 \pmod N$, которое при малых значениях a позволяет находить равенства (3). При этом, если длина интервала для квадратичного решета равна l, то для значений β верна оценка сверху $\beta < al\sqrt{N}$.

Основной проблемой при реализации MPQS-алгоритма является необходимость частого (при каждой смене многочлена) вычисления корней многочленов по модулю маленьких простых. Для снижения трудоемкости решения этой проблемы в 1993 году был предложен моди-

фицированный алгоритм HMPQS (см. статьи [7, 8]), использующий многочлены вида

$$f(x) = t^2x^2 + 2sx + \frac{s^2 - N}{t^2}, \quad \text{где } s^2 \equiv N \pmod{t^2}.$$

Появление в начале 90-х годов алгоритма решета числового поля (NFS) для решения задачи факторизации и доказательство для него ассиптотически наилучшей оценки трудоемкости практически прекратило дальнейшие исследования в области алгоритмов квадратичного решета.

В настоящем докладе мы излагаем одну из возможных модификаций MPQS-алгоритма, которая обладает рядом возможностей, позволяющих снизить трудоемкость алгоритма при практической реализации, а именно

- генерация многочленов, используемых для просеивания и поиска соотношений вида (2), производится по итерационной формуле, т. е. $f_{n+1} = \mathcal{F}(f_n)$;
- предлагается критерий для простого выбора начального многочлена $f_0(x)$;
- величины β, являющиеся значениями многочленов в целых точках, ограничены сверху: $\beta < R\sqrt{N}$, где R некоторая постоянная, зависящая от длины интервала просеивания;
- для любого простого числа p решения уравнения $f_{n+1}(x) \equiv 0 \pmod{p}$ могут быть линейно выражены через решения уравнения $f_n(x) \equiv 0 \pmod{p}$, $n = 0, 1, \ldots$

Добавим, что первоначальный вариант данного алгоритма (без выбора начального многочлена), был предложен автором в 1997 году. Кроме того, все перечисленные ранее в докладе алгоритмы, включая алгоритм Моррисона и Бриллхарта, при определенном выборе параметров, являются частными случаями излагаемого далее алгоритма.

2. Свойства «подходящих» многочленов

Приведем полученные теретические результаты.

Определение 1. Мы будем называть произвольный многочлен второй степени $f(x) \in \mathbb{Z}[x]$, $f(x) = ax^2 + bx + c$, $a \neq 0$, «подходящим», если a является квадратичным вычетом по модулю N и для дискриминанта D_f выполнено условие

$$D_f = b^2 - 4ac \equiv 0 \pmod{N}. \tag{4}$$

Мы будем считать, что выполнены равенства

$$\gcd(a, N) = 1, \quad \gcd(b, N) = 1, \quad \gcd(c, N) = 1.$$

Более того, из равенства $D_f = kN$ следует, что для $d = \gcd(a, b, c)$ верна оценка $d < \sqrt{k}$. В противном случае получаем $d^2m = kN$, для некоторого целого числа m, и $d^2 \mid N$.

Приведенные выше условия позволяют нам сформулировать следующую лемму.

Лемма 1. *Пусть $f(x) \in \mathbb{Z}[x]$, $f(x) = ax^2 + bx + c$ подходящий многочлен. Тогда для любого α найдется такое γ, и наоборот, для любого γ найдется такое α, что*

$$f(\alpha) \equiv \gamma^2 \pmod{N}.$$

Расмотрим произвольную последовательность целых чисел $\{\alpha_n\}$, $\alpha_n \neq 0$, $n = 0, 1, \ldots$, которую в дальнейшем мы будем называть «управляющей последовательностью», и определим последовательность многочленов $\{f_n(x)\}$, $f_n(x) \in \mathbb{Z}[x]$, где $f_0(x)$ подходящий многочлен, следующим образом

$$\begin{aligned} f_n(x) &= a_n x^2 + b_n x + c_n, \quad a_n, b_n, c_n \in \mathbb{Z}, \\ f_{n+1}(x) &= x^2 f_n\left(\alpha_n + \frac{1}{x}\right), \quad n = 0, 1, \ldots . \end{aligned} \tag{5}$$

Рассмотрим еще одну последовательность целых чисел $\{\gamma_n\}$, $n = -1, 0, 1, \ldots$, зависящую от $\{\alpha_n\}$, следующим образом

$$\begin{aligned} &\gamma_{-1} \text{ удовлетворяет сравнению } \gamma_{-1}^2 \equiv a_0 \pmod{N}, \\ &\gamma_0 \equiv \frac{2a_0\alpha_0 + b_0}{2\gamma_{-1}} \pmod{N}, \\ &\gamma_{n+1} = \alpha_{n+1}\gamma_n + \gamma_{n-1}, \quad n = 0, 1, \ldots \end{aligned} \tag{6}$$

Верна следующая лемма.

Лемма 2. *Пусть $\{\alpha_n\}$, $\{f_n(x)\}$ и $\{\gamma_n\}$ определенные выше последовательности. Тогда для любого индекса $n = 0, 1, \ldots$ выполнены следующие утверждения*

1. *Дискриминанты всех многочленов совпадают, то есть*

$$D_{f_{n+1}} = D_{f_n},$$

2. *Многочлен $f_n(x)$ является подходящим многочленом,*
3. *Выполнено сравнение*

$$f_n(\alpha_n) \equiv \gamma_n^2 \pmod{N}.$$

Последняя лемма позволяет нам построить последовательность подходящих многочленов, каждый из которых может быть использован в алгоритме квадратичного решета для поиска соотношений вида (2).

3. Алгоритм

Приведем схематичное описание нашего алгоритма факторизации.

1. Выберем начальный многочлен $f_0(x) \in \mathbb{Z}$, начальные значения α_0, γ_{-1}, γ_0 и множество простых чисел p, составляющих факторную базу $\mathcal{B}$.
2. Определим интервал просеивания $\mathcal{J}$, зависящий от коэффициентов многочлена.
3. Выполним алгоритм квадратичного решета на интервале $\mathcal{J}$. Определим значения β и, используя (6), соответствующие им значения γ, для которых выполнены условия (2) и (3).
4. Если набранных соотношений достаточно, то перейдем к шагу 7.
5. Определим новое значение α_n и, используя формулы (5), вычислим новый подходящий многочлен $f_n(x)$
6. Вычислим значения корней уравнения $f_n(x) \equiv 0 \pmod{p}$ для всех простых $p \in \mathcal{B}$ и перейдем к шагу 2.
7. Выберем из множества пар (β, γ) те, которые дадут нам сравнение (1) и делитель числа N.

Далее мы изложим некоторые принципы реализации перечисленных выше шагов алгоритма факторизации.

4. О выборе начального многочлена

В своей диссертации [3] Х. Бендер предложил критерий для выбора многочленов, используемых в алгоритме квадратичного решета, который в последствии был обобщен Б. Мёрфи [6] на случай решета числового поля. Мы приведем вариант данного критерия в формулировке Мёрфи.

Рассмотрим факторную базу алгоритма факторизации $\mathcal{B}$, т. е. множество простых чисел p, участвующих в разложении (3), включая $p = 2$. Тогда определим функцию

$$\varkappa(f) = \sum_{p \in \mathcal{B}} (1 - q_p) \frac{\log p}{p - 1}, \quad \varkappa(f) \in \mathbb{R},$$

где

$$q_p = \begin{cases} 2, & \text{если } \left(\dfrac{D_f}{p}\right) = 1, \\ 0, & \text{если } \left(\dfrac{D_f}{p}\right) = -1,\ 0. \end{cases}$$

Для $p = 2$ величина $q_p = 2$, если хотя бы один из коэффициентов a, c нечетен, в противном случае $q_p = 0$. Обозначим $P(f)$ вероятность того, что при случайном выборе целого x значение $\beta = f(x)$ удовлетворяет (3) для всех $p \in \mathcal{B}$. Тогда выполнено неравенство

$$P(f_1) > P(f_2), \quad \text{если } \varkappa(f_1) < \varkappa(f_2).$$

Таким образом, значения функции $\varkappa(f)$ позволяют ранжировать многочлены $f(x)$, используемые в алгоритме квадратичного решета и выбирать те, для которых величина $P(f)$ принимает наибольшее значение. В [3] было показано, что для многочленов, используемых в MPQS алгоритме значения функции $\varkappa(f)$ лежат в интервале $[-3.186\ldots - 0.5899]$.

Далее, мы приведем численные значения функции $\varkappa(f)$ для подходящих многочленов и определим способ отбора простых чисел в факторную базу $\mathcal{B}$. Пусть D_f дискриминант подходящего многочлена, удовлетворяющий (4). Тогда разрешимость уравнения $f(x) = ax^2 + bc + c \equiv 0 \pmod{p}$ следует из равенства

$$\left(\frac{D_f}{p}\right) = 1, \tag{7}$$

в случае, если $a \not\equiv 0 \pmod{p}$. В противном случае, когда $p \mid a$, уравнение $f(x) \equiv 0 \pmod{p}$ разрешимо либо при $b \not\equiv 0 \pmod{p}$, либо при $p \mid b$, $p \mid c$. Последний случай возможен только тогда, когда $p^2 \mid D_f$. Учитывая, что N свободно от квадратов, получаем

$$p^2 \mid k. \tag{8}$$

Таким образом, факторная база нашего алгоритма содержит только те простые, для которых выполнено условие (7) или (8). При проведении практических экспериментов на ЭВМ с 50 случайными 256-битными составными числами, мы получили, что для $D_f = kN$, $k \in \mathbb{Z}$, $1 \leqslant k \leqslant 2^{16}$, значения функции $\varkappa(f)$ лежат в интервале $[-6.005362\ldots - 5.639775]$ при достаточно больших значениях k, что значительно лучше чем в [3]. При этом значение функции $\varkappa(f)$ уменьшается с ростом k. Отметим, что из первого утверждения леммы 2 следует, что $\varkappa(f_{n+1}) = \varkappa(f_n)$, $n = 0, 1, \ldots$, где многочлены $f_n(x)$ определены по правилам (5).

5. О реализации квадратичного решета

При реализации MPQS-алгоритма для каждого многочлена необходимо вычислять значения корней по модулю всех простых чисел из факторной базы. Как показывают проведенные эксперименты, при частой смене многочленов процедура вычисления корней занимает достаточно большое количество времени и сильно увеличивает на практике время работы алгоритма в целом. В связи с этим мы приводим простой

результат, позволяющий существенно снизить трудоемкость нахождения корней.

Лемма 3. *Пусть* $\{f_n(x)\}$, $n = 0, 1, \ldots$ *определенная выше последовательность подходящих многочленов. Пусть* p *произвольное простое число и* $\delta_n^{1,2}$ *решения уравнения* $f_n(x) \equiv 0 \pmod{p}$. *Тогда для решений* $\delta_{n+1}^{1,2}$ *уравнения* $f_{n+1}(x) \equiv 0 \pmod{p}$, *при* $a_{n+1} \not\equiv 0 \pmod{p}$, *выполнено сравнение*

$$\delta_{n+1}^{1,2} \equiv \frac{a_n(\delta_n^{1,2} - \alpha_n)}{a_{n+1}} \pmod{p}.$$

Из утверждения приведенной выше леммы следует, что корни следующего подходящего многочлена линейно выражаются через корни предыдущего многочлена, что позволяет на практике существенно снизить время работы алгоритма.

6. Заключение

В докладе мы не указали способ выбора управляющей последовательности α_n. Исходя из полученных автором оценок на значения многочленов $f_n(x)$, можно высказать следующее гипотетическое предположение: наилучшим выбором значения α_n является выбор такого целого числа, при котором $|f_n(x)|$ принимает наименьшее значение, т. е.

$$\alpha_n = \{x \in \mathbb{Z} \colon \forall\, y \in \mathbb{Z},\ y \neq x \implies |f_n(y)| \geqslant |f_n(x)|\}.$$

В настоящее время автором проводятся активные эксперименты на ЭВМ, направленные на практическое подтверждение указанной гипотезы.

Список литературы

[1] *Василенко О. Н.* Теоретико-числовые алгоритмы в криптографии. М.: МЦНМО, 2003.
[2] *Кнут Д.* Искусство программирования. Получисленные алгоритмы. Том II, 3-е издание. М.: 2000.
[3] *Boender H.* The number of relations in the quadratic sieve algorithm. Chapter 4. PhD Thesis, University of Leiden, 1997.
[4] *Kraitchik M.* Theorie des Nombers. Tome II. Gautier-Villars, Paris, 1926.
[5] *Morrison M., Brillhart J.* A method of factoring and factorization of F_7. Mathematics Of Computation, Vol. 29, № 129, 1975.
[6] *Murphy B., Brent R. P.* On quadratic polynomials for the number field sieve. Report of The Australian National University, August, 1997.

[7] *Peralta R.* A quadratic sieve on the n-dimensional cube. CRYPTO'92, Vol. 740 of Lecture Notes in Computer Science, Springer-Verlag, pp. 324–332, 1993.
[8] *Peralta R.* Implementation of the hybercube multiple polynomial quadratic sieve. Preprint.
[9] *Pomerance C.* The quadratic sieve factoring algorithm. EUROCRYPT'84, Vol. 209 of Lecture Notes in Computer Science, Springer-Verlag, pp. 169–182, 1985.
[10] *Shanks D.* Class number, a theory of factorization and genera. Proc. Symposium Pure Math., Vol. 20, 1971.
[11] *Silverman R.* The multiple polynomial quadratic sieve. Mathematics Of Computation, Vol. 48, pp. 329–339, 1987.

О классификации обобщенных запретов всех булевых функций от трех переменных

Н. В. Никонов

Проблема определения совместности систем уравнений представляет собой важное направление прикладной дискретной математики. В докладе рассматриваются системы специального вида [1, 2], порождаемые регистром сдвига и функцией $f(\vec{x})$

$$\begin{cases} f(x_1, \ldots, x_n) = \gamma_1, \\ f(x_2, \ldots, x_{n+1}) = \gamma_2, \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ f(x_N, \ldots, x_{n+N-1}) = \gamma_N. \end{cases} \tag{1}$$

При этом комбинация знаков $\gamma_1, \ldots, \gamma_N$ получила название запрета функции $f(\vec{x})$, если для нее система (1) — несовместна. Основополагающие результаты в этой области для булевых функций были получены Сумароковым С. Н. [2]. В настоящее время теория запретов является сложившимся направлением дискретной математики. Однако в большей части опубликованных работ исследовались запреты функций с подряд идущими переменными, мало внимания уделялось проблеме их поиска и доказательству несовместности соответствующих систем уравнений.

В докладе вводится и изучается понятие обобщенного запрета $\gamma_{i_1}, \gamma_{i_1+l_1}, \ldots, \gamma_{i_1+l_1+\ldots+l_{M-1}}$, порожденного функцией $f(x_{i_1}, \ldots, x_{i_n})$, переменные x_{i_j} которой поступают с произвольных мест на регистре сдвига (см. [2])

$$\begin{cases} f(x_{i_1}, \ldots, x_{i_n}) = \gamma_{i_1}, \\ f(x_{i_1+l_1}, \ldots, x_{i_n+l_1}) = \gamma_{i_1+l_1}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ f(x_{i_1+l_1+\ldots+l_{M-1}}, \ldots, x_{i_n+l_1+\ldots+l_{M-1}}) = \gamma_{i_1+l_1+\ldots+l_{M-1}}, \end{cases} \tag{2}$$

и особое внимание уделяется проблеме доказательства того факта,

Работа выполнена при поддержке гранта Президента РФ (НШ-2358.2003.9)

что данная комбинация является запретом. В системе (2) параметры $l_1, \ldots, l_{M-1}$ определяют расстояния между знаками обобщенного запрета. Под длиной обобщенного запрета будем понимать число M — количество уравнений в системе (2).

Идея доказательства того, что выделенная комбинация знаков является запретом рассматриваемой функции, сводится к погружению множества решений исследуемой системы булевых уравнений (2) в некоторую систему линейных неравенств (3):

$$\begin{cases} a^{(1)}_{i_1} x_{i_1} + \ldots + a^{(1)}_{i_1+l_1+\ldots+l_{M-1}} x_{i_1+l_1+\ldots+l_{M-1}} \geqslant b^{(1)}, \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ a^{(k)}_{i_1} x_{i_1} + \ldots + a^{(k)}_{i_1+l_1+\ldots+l_{M-1}} x_{i_1+l_1+\ldots+l_{M-1}} \geqslant b^{(k)}. \end{cases} \tag{3}$$

Данный переход осуществляется путем замены каждого уравнения системы (2) на эквивалентную систему линейных неравенств [3], [4]. Очевидно, что если система (3) несовместна, то система (2) несовместна и $\gamma_{i_1}, \gamma_{i_1+l_1}, \ldots, \gamma_{i_1+l_1+\ldots+l_{M-1}}$ — запрет функции $f(x_{i_1}, \ldots, x_{i_n})$.

Условия, при которых система (3) — несовместна, могут быть использованы для построения классов k-значных функций с запретами при отображении единичного куба в куб с ребром длиной $k-1$ методом растяжения [5]. Переход к анализу k-значных преобразований отвечает устойчивым тенденциям роста объема перерабатываемой информации и повышения быстродействия соответствующих узлов дискретной аппаратуры [6]. Таким образом, каждая булевая функция от 3-х переменных с запретом порождает классы k-значных функций с запретом схожей природы, что делает актуальным составление каталога всех булевых функций от 3-х переменных.

Автором была проделана полная классификация обобщенных запретов всех функций от 3-х переменных и составлен соответствующий каталог. Построение обобщенного запрета для каждой функции представляет собой сложную аналитическую задачу, в общем случае не сводимую к переборным алгоритмам.

В докладе излагаются принципы классификации и нумерации функций каталога. Разбираются примеры функций с обобщенными запретами из составленной классификации, приводятся суммарные сведения, подводящие итог проведенных классификационных исследований.

Установлено, что для всех классов булевых функций с запретом, за исключением одного, обобщенные запреты имеют единую структуру с фиксированной длиной. Соответствующие суммарные данные приведены в таблице.

Длина обобщенного запрета $l =$	1	2	3	4	5	6	7	8	9	10	> 10 (22)
Число функций с обобщенным запретом длины l	2	24	36	40	36	–	20	28	24	4	4

Исключение составляет класс функций с запретом, линейных по средней координате, для которых необходимо проведение дополнительных исследований, отдельные результаты которых приводятся в докладе. В частности, удалось доказать, что при выполнении определенных условий, налагаемых на разности расстояний между существенными переменными данной функции, обобщенный запрет существует и построить его аналитический вид.

Список литературы

[1] *Алферов А. П.*, *Зубов А. Ю.*, *Кузьмин А. С.*, *Черемушкин А. В.* Основы криптографии. Учебное пособие. М.: Гелиос АРВ, 2001 г., стр. 266.

[2] *Сумароков С. Н.* Запреты двоичных функций и обратимость для одного класса кодирующих устройств. Обозрение прикладной и промышленной математики, том 1, выпуск 1. М.: ТВП, 1994 г., стр. 33–55.

[3] *Балакин Г. В.*, *Никонов В. Г.* Методы сведения булевых уравнений к системам пороговых соотношений. Обозрение прикладной и промышленной математики, том 1, выпуск 3. М.: ТВП, 1994 г., стр. 389–401.

[4] *Никонов В. Г.* Пороговые представления булевых функций. Обозрение прикладной и промышленной математики, том 1, выпуск 3. М.: ТВП, 1994 г., стр. 402–457.

[5] *Никонов Н. В.* Метод растяжения в изучении проблем совместности нелинейных систем уравнений. НТЖ 5, приложение 1 «Успехи современного естествознания», Украина, Крым, Ялта–Гурзуф, 18–27 мая. М.: Академия Естествознания, стр. 174–176.

[6] *Никонов В. Г.*, *Никонов Н. В.* Запреты k-значных функций и их связь с проблемой разрешимости систем уравнений специального вида. Вестник РУДН, серия «Прикладная и компьютерная математика», т. 2, 1, 2003 г., стр. 79–93.

Построение латинских квадратов в булевской параметризации и их использование в стандартах шифрования

В. А. Носов, Д. В. Алашкевич

1. Латинские квадраты в стандартах шифрования

Латинские квадраты давно нашли применение в криптографии. Согласно Шеннону [1] они представляют собой так называемый совершенный шифр. В математических моделях современных стандартов шифрования также появляются латинские квадраты над некоторым множеством. Приведем соответствующие примеры.

1.1. Криптосистема DES

DES (Data Encryption Standard) представляет собой блочный шифр, шифрующий данные 64-битовыми блоками. С одного конца алгоритма вводится 64-битовый блок открытого текста, а с другого конца выходит 64-битовый блок шифротекста. Основной цикл преобразования можно представить так

$$(a',\, b') = D_k(a,\, b) = (b,\, a \oplus f(b,\, k)),$$

где k — это 64-битный ключ, а $(a,\, b)$ — пара 64-битных блоков. Построим квадрат, характеризующий двойной цикл преобразований, то есть построенный на основания функции $D_{k_2} D_{k_1}(a,\, b)$. Пара $(k_1,\, k_2)$ будет задавать номер строки, а пара $(a,\, b)$ — номер столбца. В клетке будет стоять пара $D_{k_2} D_{k_1}(a,\, b)$. Тогда нетрудно доказать, что квадрат будет латинским если функция f — биекция. Заметим, что в реальной системе DES функция f небиективна.

1.2. Криптосистема IDEA

Криптоалгоритм IDEA (International Data Encryption Algorithm) представляет собой вариант 64-битного итеративного блочного шифра с 128-битным ключом и восемью циклами криптографического преобразования. Секретность преобразования обеспечивается за счет трех различных типов арифметических операций над 16-битными словами: $\oplus$ —

поразрядного сложения, $+$ — сложения по модулю 2^{16} и $*$ — умножения по модулю $2^{16}+1$. В отличие от DES здесь шифруется не пара блоков, а четверка. Основной цикл преобразований описывается так

$$\begin{aligned} A' &= A \oplus \big((B \oplus D + (A \oplus C) * K_1) * K_2\big), \\ B' &= C \oplus \big((B \oplus D + (A \oplus C) * K_1) * K_2\big), \\ C' &= B \oplus \big((A \oplus C) * K_1 + ((B \oplus D + (A \oplus C) * K_1) * K_2)\big), \\ D' &= D \oplus \big((A \oplus C) * K_1 + ((B \oplus D + (A \oplus C) * K_1) * K_2)\big). \end{aligned} \tag{1}$$

Если зафиксируем $A = B = C = K_1 = 0$, то получим соотношение для D':

$$D' = D \oplus D * K_2. \tag{2}$$

Поскольку $2^{16}+1$ — это простое число (четвертое число Ферма), то если известно D' и D — K_2 восстанавливается однозначно. Аналогично, если известно D' и K_2 — D находится однозначно. Следовательно, квадрат, в котором по номеру строки K_2 и номеру столбца D однозначно определяется элемент D' из соотношения (2), будет латинским.

2. Латинские квадраты и правильные семейства

В работе [2] был предложен способ построения параметрических классов латинских квадратов размера $2^n \times 2^n$ над множеством булевых векторов длины n. Данная конструкция реализуется с использованием семейств булевых функций, обладающих свойством *правильности*.

$$\begin{cases} f_1 = x_1 + y_1 + g_1(\pi_1(x_1, y_1), \ldots, \pi_n(x_n, y_n)), \\ f_2 = x_2 + y_2 + g_2(\pi_1(x_1, y_1), \ldots, \pi_n(x_n, y_n)), \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ f_n = x_n + y_n + g_n(\pi_1(x_1, y_1), \ldots, \pi_n(x_n, y_n)). \end{cases} \tag{3}$$

Теорема 1. *Система булевых функций вида* (3) *определяет латинский квадрат для любых функций от двух переменных* $\pi_1, \ldots, \pi_n$ *тогда и только тогда, когда семейство функций* $g = (g_1, \ldots, g_n)$ *является правильным.*

Данная конструкция дает широкий класс латинских квадратов. Найден ряд критериев и алгоритмов проверки правильности семейств. Для широкого класса функций описаны типы правильных семейств. В то же время задача перечисления правильных семейств является нерешенной.

В работе [3] получены критерии правильности для семейства булевых функций. Приведем один из этих критериев, на языке существенной зависимости функций от некоторого подмножества переменных.

Теорема 2. *Семейство булевых функций* $f = (f_i)$, $i \in [1, n]$ *будет правильным тогда и только тогда, когда для любого подмножества* I, $I \subseteq [1, n]$ *произведение функций* $\prod_{i \in I} f_i$ *не зависит существенно от множества переменных* $x_I = (x_i)$, $i \in I$.

Получены достаточные условия правильности семейства булевых функций в терминах графа существенной зависимости переменных.

Теорема 3. *Пусть* $f = (f_1, \ldots, f_n)$ *— семейство булевых функций,* G_f *— его граф существенной зависимости переменных. Если для любого простого элементарного цикла* C *графа* G_f *выполнено*

$$\prod_{i \in C} f_i \equiv 0, \tag{4}$$

то семейство f *является правильным.*

Найдены необходимые и достаточные условия для правильности семейств линейных булевых функций.

Утверждение 1. *Произвольное семейство линейных булевых функций* $f = (f_1, \ldots, f_n)$ $(f_i \in L \ \ \forall\, i \in \overline{1, n})$ *является правильным тогда и только тогда, когда после перенумерации индексов приводится к треугольному виду.*

3. Ортогональные латинские квадраты

Приведена конструкция, позволяющая строить тройки попарно ортогональных латинских квадратов. Один из которых имеет вид $f_i = x_i + y_i$, $i \in \overline{1, n}$. Другие строятся с помощью специальной системы функций

$$(f_1, \ldots, f_n), \quad f_i = f_i(x_1, \ldots, x_n), \quad i \in \overline{1, n}.$$

Утверждение 2. *Пусть семейство функций* $(f_1, \ldots, f_n)$ *обладает следующими свойствами: семейства булевых функций* $(f_1, \ldots, f_n)$ *и* $(f_1 + x_1, \ldots, f_n + x_n)$ *являются регулярными. Тогда квадраты, заданные семействами* $\{x_i + y_i\}$, $\{f_i(x_i + y_i) + x_i\}$ *и* $\{f_i(x_i + y_i) + y_i\}$ *будут задавать* 3 *попарно ортогональных латинских квадрата.*

Список литературы

[1] *Шеннон К.* Теория связи в секретных системах. В сб. К. Шеннон. Работы по теории информации и кибернетике, М.: Издательство иностранной литературы, 1963, с. 333–369.

[2] *Носов В. А.* О построении латинских квадратов в булевской базе данных. Интеллектуальные Системы, М.: 1999, т. 4, вып. 3–4, с. 307–320.

[3] *Носов В. А.* Критерий регулярности булевского неавтономного автомата с разделенным входом. Интеллектуальные Системы, М.: 1998, т. 3, вып. 3–4, с. 269–280.

[4] *Denes J., Keedwell A. D.* Latin squares and their applications. Budapest, 1974.

Асимптотическая формула для числа (n, m, k)-устойчивых функций

К. Н. Панков

Пусть $V_k = \{x_1, \ldots, x_k\}$, $x_i \in \{0, 1\}$ — множество двоичных векторов длины k. Множество всех отображений из V_n в V_m будем обозначать $F_{n,m}$. Можно рассматривать значение функции $f \in F_{n,m}$ как вектор длины m, компонентами которого являются значения ее координатных двоичных функций от n переменных:

$$f(\alpha) = (f_1(\alpha),\ f_2(\alpha),\ \ldots,\ f_m(\alpha))\colon\ V_n \to V_m,$$

где $f_i(\alpha)\colon\ V_n \to \{0,\ 1\}\ \forall\ j \in \{1,\ \ldots,\ m\}$

Функция $f \in F_{n,m}$ называется (n, m, k)-устойчивой, если для любых наборов $1 \leqslant i_1 < \ldots < i_k \leqslant n$, $a_j \in \{0,\ 1\}\ \forall\ j \in \{1,\ \ldots,\ k\}$ отображение $f' = \left((f_1)^{a_1,\ \ldots,\ a_k}_{i_1,\ \ldots,\ i_k},\ \ldots,\ (f_m)^{a_1,\ \ldots,\ a_k}_{i_1,\ \ldots,\ i_k}\right) \in F_{n-k,m}$ является уравновешенным. Функция от $n-k$ переменных $g^{a_1,\ \ldots,\ a_k}_{i_1,\ \ldots,\ i_k}$ принимает значения, равные значениям функции g на векторах из V_n у которых координаты с номерами $i_1,\ \ldots,\ i_k$ равны соответственно $a_1,\ \ldots,\ a_k$. Данная функция является уравновешенной, если

$$\forall\ \overline{v} \in V_m\quad \left|\left\{\overline{u} \in V_{n-k}\colon\ g^{a_1,\ \ldots,\ a_k}_{i_1,\ \ldots,\ i_k}(\overline{u}) = \overline{v}\right\}\right| = 2^{n-k-m}.$$

В ряде работ (см. обзор [5]–[9]) изучались свойства функций из класса $R(n, m, k)$ всех (n, m, k)-устойчивых функций. В монографии [4] доказано, что функция $f \in F_{n,m}$ является (n, m, k)-устойчивой тогда и только тогда, когда любая ненулевая линейная комбинация ее координатных функций является (n, s, k)-устойчивой при $s = 1$.

Для произвольного непустого подмножества $J \in \{j_1,\ \ldots,\ j_{|J|}\}$ множества $\{1,\ \ldots,\ m\} = \overline{1,\ m}$ и произвольного подмножества $I \in \{i_1,\ \ldots,\ i_{|I|}\}$ множества $\{1,\ \ldots,\ n\} = \overline{1,\ n}$ через $w^J_I(f)$ обозначим вес $\left\|(f^J)^{1,\ \ldots,\ 1}_{i_1,\ \ldots,\ i_{|I|}}\right\|$ подфункции $(f^J)^{1,\ \ldots,\ 1}_{i_1,\ \ldots,\ i_{|I|}}$ функции $f^J = f_{j_1} \oplus \ldots \oplus f_{j_{|J|}}$ на векторах, у которых координаты с номерами $i_1,\ \ldots,\ i_{|I|}$ равны единице. Согласно [1] f^J является $(n, 1, k)$-устойчивой тогда и только тогда, когда $w^J_I(f) = 2^{n-|I|-1}$ $\forall\ I \subset \overline{1,\ n}\colon\ 0 \leqslant |I| \leqslant k$.

Пусть далее функция f выбирается случайно и равновероятно из множества $F_{n,m}$. Тогда

$$\frac{|R(n, m, k)|}{|F_{n,m}|} = \Pr\{f \in R(n, m, k)\} = \Pr\{\overline{w} = E\overline{w}\},$$

где $\overline{w} = \overline{w}(f, n, m, k) = \left(w^J_I(f),\ 0 \leqslant |I| \leqslant k,\ I \subset \overline{1,\ n},\ \varnothing \neq J \subset \overline{1,\ m}\right)$ — случайный вектор длины $L = L(n, m, k) = \sum_{s=1}^{m} \binom{m}{s} \sum_{i=0}^{k} \binom{n}{i}$, $\mathsf{E}\overline{w}$ — математическое ожидание случайного вектора $\overline{w}$.

Теорема. *Пусть фиксировано $m \in \mathbb{N}$, $n \to \infty$, $k(n) = o(\sqrt{n})$, $Q = Q(n, m, k)$ — ковариационная матрица случайного вектора $(\overline{w} - \mathsf{E}\overline{w})/2^{n/2+m-2}$, $\left\{\overline{z}(n)2^{n/2+m-2}\right\}_{n=1}^{\infty}$ — последовательность целочисленных векторов размерности $L(n, m, k)$. Тогда равномерно относительно $\overline{z}(n)$:*

$$\Pr\left\{\overline{w} = \mathsf{E}\overline{w} + \overline{z}(n)2^{\frac{n}{2}+m-2}\right\} = \frac{\exp\left(-\frac{1}{2}\overline{z}^T Q^{-1}\overline{z}\right) + O\left(\frac{n^{4k+3}}{2^{n/2}}\right)}{\left(2^{\frac{n}{2}-1}\right)^L \sqrt{(2\pi)^L \det Q}} =$$

$$= \frac{\exp\left(-\frac{1}{2}\sum\limits_{J \subset \overline{1,m}}\ \sum\limits_{\substack{I \subset \overline{1,m} \\ |I| \leqslant k}} \left(2^{|I|+m-1} z^J_I - \sum\limits_{i \in I} 2^{|I|+m-2} z^J_{I \setminus \{i\}}\right)^2\right) + O\left(\frac{n^{4k+3}}{2^{n/2}}\right)}{\exp\left(\left(\frac{n-k}{2}\binom{n}{k}(2^m - 1) - L(n, m, k)\left(m - \log_2\sqrt{2\pi}\right)\right)\ln 2\right)}. \tag{1}$$

В качестве следствия из нее получаем, что при $m \in \mathbb{N}$, $n \to \infty$, $k(n) = o(\sqrt{n})$

$$|R(n, m, k)| \sim \exp\left(\left(m2^n - \frac{n-k}{2}\binom{n}{k}(2^m - 1) + \right.\right.$$
$$\left.\left. +\ L(n, m, k)\ \left(m - \log_2\sqrt{2\pi}\right)\right)\ln\ 2\right). \tag{2}$$

Аналогичные результаты для случая $m = 1$ были получены в [2].

Список литературы

[1] *Денисов О. В.* Асимптотическая формула для числа корреляционно-иммунных порядка k булевых функций. Дискретная математика, т. 3, вып. 2 (1991).

[2] *Денисов О. В.* Локальная предельная теорема для распределения части спектра случайной двоичной функции. Дискретная математика, т. 12, вып. 1 (2000).

[3] *Кузнецов Ю. В.*, *Шкарин С. А.* Коды Рида — Маллера (обзор публикаций). Математические вопросы кибернетики, вып. 6 (1996).
[4] *Логачев О. А.*, *Сальников А. А.*, *Ященко В. В.* Булевы функции в теории кодирования и криптологии. М.: МЦМНО, 2004.
[5] *Тарannиков Ю. В.* О корреляционно-иммунных и устойчивых булевых функциях. Математические вопросы кибернетики, вып. 11 (2002).
[6] *Cheon J. H.* Nonlinear Vector Resilient Function. Crypto'2001, Springer-Verlag (2001), pp. 458–469.
[7] *Bierbrauer J.*, *Gopalacrishnan K.*, *Stinson D.* Bounds on Resilient Function and Orthogonal Arrays. Crypto'94, Springer-Verlag (1994), pp. 247–256.
[8] *Gopalacrishnan K.*, *Stinson D.* Three Characterization of Non-binary Correlation-Immune and Resilient Function. 1997, `www.eprint.iacr.org`.
[9] *Gupta K. C.*, *Sarcar P.* Improved Construction of Nonlinear Resilient S-Boxes. Asiacrypt'2002, Springer-Verlag (2002), pp. 466–483.

Об алгоритме поточного шифрования VR

М. А. Пудовкина

Начиная с 1996 года предлагаются разные модификации алгоритма шифрования RC4. Среди них наиболее известнымы являются алгоритмы поточного шифрования ISAAC, IBAA, IA [1], обобщением которых является алгоритм GI [2]. В работе [3] предложена еще одна модификация алгоритма RC4 — алгоритм поточного шифрования VMPC. Предложенный алгоритм VMPC, является частным случаем алгоритма VR, описание которого приведем.

Алгоритм $\mathrm{VR} = \mathrm{VR}(m,\ r,\ k,\ l,\ (\alpha_1,\ \ldots,\ \alpha_r),\ (\beta_1,\ \ldots,\ \beta_r))$ зависит от параметров $m = 2^n$, натуральных чисел $n \geqslant 2$, r, k, l, $l \leqslant k < r$, векторов $(\alpha_1,\ \ldots,\ \alpha_r)$, $(\beta_1,\ \ldots,\ \beta_r)$ из $Z^r_{\varphi(m)}$. Алгоритм VR моделируется автономным автоматом $A_{\mathrm{VR}} = (Z_m \times Z_m \times S_m,\ Z_m,\ Z_m,\ F_{\mathrm{VR}},\ f_{\mathrm{VR}})$, где

$$F_{\mathrm{VR}} \colon Z_m \times Z_m \times S_m \to Z_m \times Z_m \times S_m, \quad f_{\mathrm{VR}} \colon Z_m \times Z_m \times S_m \to Z_m.$$

Состоянием алгоритма в такте $t \geqslant 1$ является тройка

$$v_t = (i_t,\ j_t,\ s_t) \in Z_m \times Z_m \times S_m.$$

Пусть $g \colon i \to i + 1 \pmod m$ — полноцикловая подстановка из S_m и пусть подстановка $\mu((\alpha_1,\ \ldots,\ \alpha_l),\ (\beta_1,\ \ldots,\ \beta_l))$: $s \to s^{\alpha_1} g^{\beta_1} \ldots s^{\alpha_l} g^{\beta_l}$ действует на S_m, $\mu_s((\alpha_1,\ \ldots,\ \alpha_l),\ (\beta_1,\ \ldots,\ \beta_l)) = s\mu((\alpha_1,\ \ldots,\ \alpha_l),\ (\beta_1,\ \ldots,\ \beta_l))$. Приведем описание t-го ($t = 1,\ 2,\ \ldots$) такта работы алгоритма VR.

Функция переходов F_{VR}

1) $i_t = i_{t-1} + 1 \pmod m$;
2) $j_t = i_t \mu_{s_t}((\alpha_1,\ \ldots,\ \alpha_k),\ (\beta_1,\ \ldots,\ \beta_k))$, $\beta_l = j_{t-1}$;
3) $s_{t+1}[i_t] = s_t[j_t]$, $s_{t+1}[j_t] = s_t[i_t]$, $s_{t+1}[k] = s_t[k]$ при $k \notin \{i_t,\ j_t\}$.

Функция выходов f_{VR}

$$z_t = j_t \mu_{s_t}((\alpha_{k+1},\ \ldots,\ \alpha_r),\ (\beta_{k+1},\ \ldots,\ \beta_{r-1},\ 0)).$$

В алгоритме VR значения параметров выбраны следующим образом: $\alpha_1 = 1,\ \alpha_2 = 1,\ \alpha_3 = 1,\ \alpha_4 = 1,\ \alpha_5 = 2,\ \beta_1 = j_{t-1},\ \beta_2 = 0,\ \beta_3 = 1,\ k = 2,\ r = 4$, т. е.

$$j_t = i_t s_t g^{j_{t-1}} s_t, \quad z_t = j_t s_t g s_t^2.$$

Очевидно, что

$$\bigcup_{l=1}^{\infty} \bigcup_{\substack{(\alpha_1,\,\ldots,\,\alpha_l),\\(\beta_1,\,\ldots,\,\beta_l)}} s\mu((\alpha_1,\,\ldots,\,\alpha_l),\,(\beta_1,\,\ldots,\,\beta_l)) = \langle g,\,s\rangle.$$

Утверждение 1. *Пусть g — произвольная фиксированная полноцикловая подстановка из S_m, подстановка s выбирается случайно и равновероятно из S_m, $m = 2^n$ и $n-1$ — составное число. Тогда вероятность того, что группа $\langle g,\,s\rangle$ изоморфна S_m, равна $\sum_{k=1}^{n-1}(2^{n-k}!-1)^{2^k}/(2^n!)$.*

Утверждение 2. *Пусть i — произвольное фиксированное число из Z_m; $\beta \in \{\overline{0,\,\varphi(m)-1}\}$, $\alpha_1,\,\alpha_2 \in \{\overline{1,\,\varphi(m)-1}\}$; подстановка s случайно и равновероятно выбирается из S_m и случайная величина $\xi_2(i,\,\beta,\,\alpha_1,\,\alpha_2)$ есть $\xi_2(i,\,\beta,\,\alpha_1,\,\alpha_2) = \xi = is^{\alpha_1}g^{\beta}s^{\alpha_2}$. Тогда для любого $z \neq ig^{\beta}$ из Z_m выполняется неравенство*

$$\Pr\{\xi_2 = ig^{\beta}\} - \Pr\{\xi_2 = z\} \geqslant \frac{1}{m(m-1)}.$$

Кроме того, при $\alpha_1 = \alpha_2$

$$\Pr\{\xi_2 = ig^{\beta}\} - \Pr\{\xi_2 = z\} \geqslant \frac{m-\alpha_1+\varphi(\alpha_1)-3}{m(m-1)}.$$

В работах [4, 5, 6] показано, что распределение первых знаков гаммы RC4 является неравномерным. Из утверждения 2 следует, если в алгоритме VMPC подстановка s_0 выбирается случайно и равновероятно из S_m, $j_0 = 0$, и i_0 — произвольное фиксированное число из Z_m, то первый знак гаммы z_1 не является равномерно распределенным. Таким образом, предложенная модификация алгоритма RC4 также обладает этой слабостью.

Список литературы

[1] *Jenkins R. J.* ISAAC. Fast Software Encryption'96, LNCS, Springer, 2004.

[2] *Погорелов Б. А., Пудовкина М. А.* О свойствах криптоалгоритма GI. Материалы конференции МаБИТ'03 (МГУ, 2003).

[3] *Zoltak B.* WMPC one-way function and stream cipher. Fast Software Encryption'2004, LNCS, Springer, 2004.

[4] *Pudovkina M.* Statistical weaknesses in the alleged RC4 keystream generator. 4nd International Workshop on Computer Science and Information Technologies (CSIT'2002), Greece, Patras, 2002.

[5] *Fluhrer S., McGrew D.* Statistical Analysis of the alleged RC4 keystream generator. Fast Software Encryption'2000, LNCS, Springer, 2000.

[6] *Mantin I., Shamir A.* A practical attack on broadcast RC4. Fast Software Encryption'2001, LNCS, Springer, 2001.

Новый класс статистических тестов для случайных чисел и его применение в задачах криптографии

Б. Я. Рябко, В. А. Монарев, Ю. И. Шокин

1. Введение

В криптографических системах защиты информации случайные и псевдослучайные числа находят самое широкое применение, поэтому задачи построения надежных генераторов и статистических тестов для их проверки привлекают внимание многих исследователей, см., например, [2], [12]. О важности для криптографии задачи тестирования случайных чисел свидетельствует, в частности, реализация Национальным бюро стандартов США проекта по отбору надежных статистических тестов для проверки генераторов случайных чисел, используемых в криптографических системах защиты информации [6].

Среди других применений статистических тестов в криптографии мы отметим известную проблему различения зашифрованных сообщений и случайных последовательностей, представляющую значительный интерес, в частности, для стеганографии [12], [10]. Трудность построения таких тестов объясняется тем, что даже заведомо «неслучайные» данные (скажем, тексты на русском языке) современные блоковые шифры должны зашифровывать в последовательности, статистически неразличимые от получаемых при подбрасывании симметричной монеты (стороны которой помечены нулем и единицей). В частности, это требование предъявлялось к алгоритмам, участвовавшим в недавно проводимом американском конкурсе на новый стандарт блокового шифра [3].

Формально статистический тест для проверки «случайных» чисел описывается следующим образом: рассматривается основная статистическая гипотеза H_0, о том, что случайная последовательность $x_1x_2\ldots x_t$ букв из алфавита $\{0,\,1\}$ порождена Бернуллиевским источником, против альтернативной гипотезы H_1, что эта последовательность порожде-

Работа поддержана Российским фондом фундаментальных исследований (грант 03-01-00495), INTAS (Grant 00-738) и Royal Society, UK (grant ref: 15995).

на стационарным и эргодическим источником (отличным от источника при выполнении H_0).

В докладе мы предлагаем два статистических теста для решения данной задачи и экспериментально сравниваем их с 16 методами, рекомендованными Американским институтом стандартов (NIST) в для криптографических приложений [6]. Показано, что новые тесты превосходят методы из [6]. Отметим, что предлагаемые тесты базируются на идеях и методах теории информации и развивают результаты авторов, анонсированные в [9].

2. Описание тестов

В этом параграфе мы дадим описание двух тестов, базирующихся на идеях универсального кодирования.

Первый из них назван «стопка книг», второй — «порядковый тест». Для их описания нам понадобятся некоторые определения. Пусть $A = \{a_1, \ldots, a_S\}$, — алфавит и пусть дан некоторый источник информации, порождающий буквы из этого алфавита. Мы рассмотрим две следующие гипотезы: H_0 — источник порождает все символы независимо и с равными вероятностями ($p(a_1) = \ldots = p(a_S) = 1/S$) и $H_1 = \neg H_0$. Необходимо построить тест для проверки гипотезы H_0 против альтернативной гипотезы H_1 по выборке $x_1x_2\ldots x_n$, $n \geqslant 1$. При применении теста «стопка книг» все буквы алфавита A упорядочиваются и нумеруются числами от 1 до S. После анализа каждой буквы x_t из $x_1x_2, \ldots, x_n$ данный порядок на множестве букв изменяется по формуле

$$\nu^{t+1}(a) = \begin{cases} 1, & \text{если } x_t = a; \\ \nu^t(a) + 1, & \text{если } \nu^t(a) < \nu^t(x_t); \\ \nu^t(a), & \text{если } \nu^t(a) > \nu^t(x_t), \end{cases} \qquad (1)$$

где ν^t — порядок после просмотра $x_1x_2, \ldots, x_t$, $t = 1, \ldots, n$, ν^1 определяется произвольно. (Например, его можно задать как $\nu^1 = \{a_1, \ldots, a_S\}$.)

Поясним сначала неформально (1). Предположим, что буквы алфавита расположены как книги в стопке и $\nu^1(a)$ — позиция в «стопке». Пусть первая буква x_1 слова $x_1x_2\ldots x_n$ будет a. Если она находится на i_1-ом месте в стопке, т. е. ($\nu^1(a) = i_1$), то a извлекается из стопки и помещается на верх. (Фактически это эквивалентно изменению нумерации в соответствии с (1).) Эта же последовательность действий повторяется с x_2 и т. д.

Основная идея этого преобразования проста — если гипотеза H_1 выполняется, то те буквы из A, частота которых выше $1/S$, будут проводить больше времени в верхней части стопки и будут иметь относительно небольшие номера. Если же H_0 выполняется, то вероятность найти любую букву x_i на любом месте в стопке книг равна $1/S$.

При применении теста множество всех номеров (или позиций в стопке книг) $1, \ldots, S$ разбивается на r, $r \geqslant 2$, подмножеств $A_1 = \{1, 2, \ldots, k_1\}$, $A_2 = \{k_1+1, \ldots, k_2\}, \ldots, A_r = \{k_{r-1}+1, \ldots, k_r\}$. Затем по выборке $x_1x_2\ldots x_n$ вычисляем количество номеров $\nu^t(x_t)$, $t = 1, \ldots, n$, принадлежащих подмножествам A_k, $k = 1, \ldots, r$. Мы обозначим это число через n_k (формально, $n_k = |\{t : \nu^t(x_t) \in A_k, \ \ t = 1, \ldots, n\}|$, $k = 1, \ldots, r$). Очевидно, если H_0 выполняется, то вероятность того, что «номер» $\nu^t(x_t)$ принадлежит A_k равна $|A_j|/S$. Затем, используя «обычный» хи-квадрат критерий, гипотеза $\widehat{H}_0 = \Pr\{\nu^t(x_t) \in A_k\} = |A_j|/S$ по «эмпирическим» частотам $n_1, \ldots, n_r$ против гипотезы $\widehat{H}_1 = \neg\widehat{H}_0$. Напомним, что вычисляется значение

$$x^2 = \sum_{i=1}^{r} \frac{(n_i - n(|A_i|/S))^2}{n(|A_i|/S)},$$

когда критерий хи-квадрат применяется и что величина x^2 асимптотически подчиняется χ^2 распределению с $(k-1)$ степенями свободы при выполнении $\widehat{H}_0$. Мы не описываем точно процедуру разбиения на подклассы $\{A_1, A_2, \ldots, A_r\}$, а предлагаем определять экспериментально это разбиение для поиска приемлимых значений, а затем проверять гипотезу по независимым данным. Этот путь вполне оправдан в случае тестирования генераторов случайных чисел, распознавания зашифрованных сообщений и многих других криптографических приложений, т. к. в этих ситуациях объем выборочных данных практически неограничен.

Рассмотрим небольшой пример, поясняющий работу алгоритма. Пусть $A = \{a_1, \ldots, a_6\}$, $r = 2$, $A_1 = \{a_1, a_2, a_3\}$, $A_2 = \{a_4, a_5, a_6\}$, $x_1\ldots x_8 = a_3a_6a_3a_3a_6a_1a_6a_1$. Если $\nu_1 = 1, 2, 3, 4, 5, 6,$ то $\nu_2 = 3, 1, 2, 4, 5, 6,$ $\nu_3 = 6, 3, 1, 2, 4, 5, \ldots$, и $n_1 = 7$, $n_2 = 1$. Мы видим, что буквы a_3 и a_6 встречаются часто и тест «стопка книг» выявляет это достаточно эффективно.

Остановимся кратко на сложности описанного теста. При «прямом» (или «наивном») преобразовании «стопки книг» в соответсвии с (1) потребовалось бы порядка S операций, что, конечно неприемлемо при больших S. Однако использование алгоритмов поиска, основанных, скажем, на известных АВЛ-деревьях, позволяет осуществить преобразование (1) за $O(\log S)$ операций.

Последнее замечание касается названия метода. В теории информации известен метод кодирования источника «стопка книг», предложенный в [7] и позже переоткрытый в [1] (См. также комментарий в [8]).

В англоязычной литературе этот метод часто называется «Move-to-Front» (MTF) схема).

Перейдем к описанию порядкового теста. Он также основан на задании нумерации $\nu^t(a)$ на буквах алфавита, однако меняющейся по правилу, отличному от (1). Для его описания мы определим величину $\lambda^{t+1}(a)$, равную количеству встреч буквы a в слове $x_1 \ldots x_{t-1} x_t$. В каждый момент времени t буквы алфавита упорядочиваются по убыванию величины $\lambda^t(a)$ и этот-то порядок и определяет $\nu^t(a)$. В остальном порядковый тест совпадает с «стопкой книг».

3. Экспериментальное сравнение предлагаемых тестов с ранее известными

Мы исследовали экспериментально описанные выше тесты и методы предложенные [6]. Выбор для сравнения методов из [6] объясняется тем, что они отобраны среди всех известных тестов на основе достаточно детального теоретического и экспериментального анализа, проведенного большим коллективом специалистов. В качестве примеров мы тестировали несколько генераторов псевдослучайных чисел, описнных

Тест / Генератор	SRG	LCG	MT19937	SEAL
Порядковый (block = 30)	19/20	0/20	0/0	0/0
Стопка книг (block = 30)	18/20	0/20	0/0	0/0
Frequency	1/0	0/0	0/0	0/0
Block Frequency	0/0	0/1	0/0	0/0
Cumulative Sums	1/2	1/0	0/0	0/0
Runs	1/0	0/0	0/0	0/1
Rank	20/20	0/0	0/0	0/0
Longest Run of Ones	1/0	0/0	1/1	0/0
Discrete Fourier Transform	1/-	3/-	0/-	0/-
Non-overlapping Templates	1/2	1/2	1/1	1/1
Overlapping Template	1/0	0/0	0/0	0/0
Universal Statistical	1/0	0/0	1/1	0/0
Approximate Entropy	3/-	4/-	2/-	3/-
Random Excursions	2/2	2/2	2/2	3/2
Random Excursions Variant	2/2	2/2	2/2	3/2
Serial	0/1	0/0	1/1	0/1
Lempel-Ziv Complexity	0/-	0/-	0/-	0/-

Таблица 1

в литературе: генератор на сдвиговых регистров (SRG) из DIEHARD (н. 13, см. `http://stat.fsu.edu/diehard/`), линейный конгруэнтный генератор (LCG) с параметрами $X_{n+1} = (134775813X_n + 1) \bmod 2^{32}$, генератор MT19937 [4], и SEAL [5]. Во всех случаях генерировалось 20 выборок и подсчитывалось количество случаев, когда тест отвергал гипотезу о случайности (H_0) при уровне значимости 0.01. Все вычисления проводились при двух длинах выборки: 2^{19} и 2^{22} бит. Данные приведены в таблице 1.

Слева от черты указано значение для длины 2^{19} бит, а справа — для 2^{22}. Например, в первом квадрате таблицы мы видим, что порядковый тест определил как не случайные 19 выборок из 20 при первой длине и 20 из 20 при второй. Прочерками в таблице указаны случаи, когда какой-либо тест не применим при данной длине.

Из таблицы мы видим, что хотя предложенные тесты «бракуют» не все генераторы, их мощность существенно выше, чем у всех методов из [6].

Список литературы

[1] *Bently J. L.*, *Sleator D. D.*, *Tarjan R. E.*, *Wei V. K.* A Locally Adaptive Data Compression Scheme. Comm. ACM, v. 29, 1986, pp. 320–330.

[2] *Menzes A.*, *van Oorschot P.*, *Vanstone S.* Handbook of Applied Cryptography. CRC Press, 1996.

[3] *Nechvatal J. and others*. Report on the Development of the Advanced Encryption Standart (AES). 2000, in: `http://csrc.nist.gov/encryption/aes/round2/r2report.pdf`.

[4] *Matsumoto M.*, *Nishimura T.* Mersenne Twister: A 623-Dimensionally Equidistributed Uniform Pseudo-Random Number Generator. ACM Transaction on Modeling and Computer Simulation, v. 8, pp. 3–30, 1998.

[5] *Rogaway P.*, *Coppersmith D.* A Software-Optimized Encryption Algorithm. In: Fast Software Encryption, Ross Anderson (Ed.), Springer-Verlag, Lecture Notes in Computer Science, v. 809, pp. 56–63, March, 1994. (see also: Journal of Cryptology, V. 11, n. 4, Autumn 1998, pp. 273–287.)

[6] *Rukhin A. and others*. A statistical test suite for random and pseudorandom number generators for cryptographic applications. NIST Special Publication 800-22 (with revision dated May 15, 2001), `http://csrc.nist.gov/rng/SP800-22b.pdf`.

[7] *Рябко Б. Я.* Сжатие данных при помощи стопки книг. Проблемы передачи информации, т. 16, № 4, 1980, с. 16–21.

[8] *Ryabko B. Ya.* A locally adaptive data compression scheme (Letter). Comm. ACM, v. 30, N 9, 1987, p. 792.

[9] *Ryabko B.*, *Monarev V.*, *Shokin Yu.* Using Universal Coding Approach to Randomness Testing. IEEE International Symposium on Information The-

ory, Proceedings, 2004, Chicago.
[10] *Ryabko B. Ya.*, *Stognienko V. S.*, *Shokin Yu. I.* A new test for randomness and its application to some cryptographic problems. Journal of Statistical Planning and Inference, 2004, v. 123, n. 2, pp. 365–376.
[11] *Рябко Б. Я.*, *Фионов А. Н.* Основы современной криптографии для специалистов в информационных технологиях. М.: Научный мир, 2004.
[12] *Schneier B.* Applied Cryptography. N. Y.: Wiley, 1996.

О построении дизъюнктных (superimposed) кодов и их использовании в криптографии

В. М. Сидельников

В работе показано, что каскадный дизъюнктный код, у которого внутреннй код является дизъюнктивным $(2, q-2)$-кодом, а внешний — произвольным q-ичным $(2, r)$-разделяющим кодом, при любых $r \geqslant 1$ и $q \geqslant 2$ является дизъюнктивным $(2, r)$-кодом (теорема 1). Эта теорема усиливает известный результат о каскадных дизъюнктных кодах. Рассмотрены некоторые приложения этого результата к криптографии.

1. Дизъюнктные коды

Набор $\mathcal{A} = \{A_1, \ldots, A_T\}$ подмножеств множества $[N] = \{1, \ldots, N\}$ называется (w, r)-семейством непокрывающих множеств (cover-free (w, r)-family), если для него выполнены следующие свойства:

$$\bigcap_{s=1}^{w} A_{i_s} \not\subseteq \bigcup_{j=1}^{r} A_{k_j} \quad \text{для всех } \{i_1, \ldots, i_w\}, \{k_1, \ldots, k_r\} \subseteq [N], \text{ таких, что } \{i_1, \ldots, i_w\} \cap \{k_1, \ldots, k_r\} = \varnothing. \tag{1}$$

Идентификационная матрица $\mathfrak{K}$ (w, r)-семейством непокрывающих множеств $\mathcal{A}$, называется дизъюнктивным (w, r)-кодом. Столбцы матрицы $\mathfrak{K}$ являются идентификационными векторами множеств A_j. Таким образом, матрица $\mathfrak{K}$ имеет N строк и T столбцов. Число N обычно называется длиной дизъюнктного кода $\mathfrak{K}$, а число T называют числом его элементов. Обычно стремятся при заданных N, (w, r) максимизировать число T. Семейство непокрывающих множеств и соответствующий ему дизъюнктный код — по существу, одно то же понятие. Эти понятия мы не будем различать.

Дизъюнктные (w, r)-коды используются для построения схем планирования экспериментов. Они находят применения и в криптографии

Работа выполнена при финансовой поддержке РФФИ (№ 02-01-00687) и NST (№ CCR-0310632) во время визита автора в Мичиганский университет летом 2004 г.

(см., например, [7, 3], а также многие другие работы). Криптографическим приложения дизъюнктных кодов рассматриваются в п. 4 данной работы.

2. Построение дизъюнктных кодов

2.1. Каскадная конструкция, использующая дизъюнктный $(2,\, q-2)$-код

Определение 1 (дизъюнктный $(2,\, q-2)$-код Q_q). Пусть

$$R_q^{(k)} = \{\{i+kq\},\, \{i+kq,\ j+kq\} \mid i,\ j=1,\ \ldots,\ q,\ \ i\neq j\},$$

$$|R_q^{(k)}| = \tbinom{q}{2} + \tbinom{q}{1} = \tbinom{q+1}{2},$$

— множество всех двух и одноэлементных подмножеств множества

$$[kq+1,\ (k+1)q] = \{kq+1,\ \ldots,\ kq+q\}.$$

Пусть $C_s^{(k)} = \{(s+kq,\ j+kq) \mid j=1,\ \ldots,\ q\}$, $|C_s^{(k)}| = q$, — множество подмножеств множества $R_q^{(k)}$, образованных множествами $\{i,\ j\} \in R_q^{(k)}$, которые содержат элемент $s+kq$.

Для каждого k семейство $\mathcal{Q}_q^{(k)} = \{C_s^{(k)} \mid s=1,\ \ldots,\ q\}$ подмножеств $\binom{q+1}{2}$-множества $R_q^{(k)}$ называется тривиальным $(2,\, q-2)$-семейством, определенном на множестве $(kq,\, (k+1)q]$. Семейство $\mathcal{Q}_q^{(0)}$ обозначается как $\mathcal{Q}_q$.

По существу, множества C_j семейства $\mathcal{Q}_q$ мы индексируем одно и двуэлементными подмножествами множества $\{1,\ \ldots,\ q\}$. Поэтому дизъюнктный код, соответствующий семейству $\mathcal{Q}_q$, имееет длину $N = \binom{q+1}{2}$ (число таких подмножеств).

Каждое семейство $\mathcal{Q}_q^{(k)} = \{C_1^{(k)}, \ldots, C_q^{(k)}\}$, $A_s^{(k)} \subset R_q^{(k)}$, содержит q элементов (подмножеств). Мощность каждого множества $C_s^{(k)}$ равна q. (Обычные обозначения: $T=q$, $N = |R_q^{(k)}| = \binom{q+1}{2}$, см. [3, 4].)

Лемма 1. *Тривиальное $(2,q-2)$-семейство $\mathcal{Q}_q^{(k)}$ является $(2,q-2)$-семейством непокрывающих множеств. Его матрица инциденций образует $(2,q-2)$-дизъюнктный код длины $N=\binom{q+1}{2}$ с числом элементов q.*

Доказательство. Очевидно,

$$C_s^{(k)} \cap C_{s'}^{(k)} = \{s+kq, s'+kq\} \text{ и } \{s+kq, s'+kq\} \cap (C_{s_1}^{(k)} \cup \ldots \cup C_{s_{q-2}}^{(k)}) \neq \varnothing, \quad (2)$$

если и только если $s \in \{s_1,\ \ldots,\ s_{q-2}\}$ или/и $s' \in \{s_1,\ \ldots,\ s_{q-2}\}$. □

Заметим, что если $k \neq k'$, то по построению $C_s^{(k)} \cap C_t^{(k')} = \varnothing$.

Если $\mathcal{A} = \{A_1,\ \ldots,\ A_q\}$, $A_j = \{a_{j,1},\ \ldots,\ a_{j,m}\} \subset [N]$, — произвольное $(2,\, q-2)$-семейство разделяющих множеств, то через $\mathcal{A}^{(k)}$ обозначается $(2,\, q-2)$-семейство $\mathcal{A}^{(k)} = \{A_1^{(k)},\ \ldots,\ A_q^{(k)}\}$ разделяющих множеств, где

$$A_j^{(k)} = \{a_{j,1} + (k-1)N,\ \ldots,\ a_{j,m} + (k-1)N\} \subset$$
$$\subset \{1 + (k-1)N,\ \ldots,\ N + (k-1)N\}.$$

Заметим, что если $k \neq k'$, то по построению $A_s^{(k)} \cap A_t^{(k')} = \varnothing$.

Определение 2 (разделяющий код). q-значный код $\mathfrak{K}$ называется разделяющим $(w,\, r)$-кодом (separating $(w,\, r)$-code), если для любых $\boldsymbol{y}_1,\ \ldots,\ \boldsymbol{y}_w \in \mathfrak{K}$ и любых $\boldsymbol{x}_1,\ \ldots,\ \boldsymbol{x}_r \in \mathfrak{K}$, $\{\boldsymbol{y}_1,\ \ldots,\ \boldsymbol{y}_w\} \cap \{\boldsymbol{x}_1,\ \ldots,\ \boldsymbol{x}_r\} = \varnothing$, существует компонента (координата) $i \in [n] = \{1,\ \ldots,\ n\}$ такая, что

$$\{y_{1,i},\ \ldots,\ y_{w,i}\} \cap \{x_{1,i},\ \ldots,\ x_{r,i}\} = \varnothing. \quad (3)$$

Определение 3 (каскадная конструкция дизъюнктного кода). Пусть $\mathfrak{K}$ — q-значный разделяющий $(2,\, r)$-код и $\mathcal{A} = \{A_1,\ \ldots,\ A_q\}$, $|A_j| = m$, — $(2,\, q-2)$-семейство разделяющих множеств, содержащее q множеств. Семейство подмножеств $\mathcal{A}_n(\mathfrak{K})$ образованное подмножествами

$$A_{\boldsymbol{x}} = A_{x_1}^{(1)} \cup A_{x_2}^{(2)} \cup \ldots \cup A_{x_n}^{(n)}, \quad \boldsymbol{x} = (x_1,\ \ldots,\ x_n) \in \mathfrak{K}, \quad |A_{\boldsymbol{x}}| = nq, \quad (4)$$

множества $\mathcal{N}_n = \cup_{k=1}^{n} \{(k-1)+1,\ \ldots,\ N+(k-1)N\} = \{1,\ \ldots,\ nN\}$, называется каскадным семейством подмножеств, порожденным разделяющим кодом $\mathfrak{K}$ и семейством $\mathcal{A}$.

Число элементов (обычно обозначаемая как T) семейства $\mathcal{A}_n(\mathfrak{K})$ равна $|\mathfrak{K}|$, число элементов $\mathcal{N}_n$ (длина соответствующего дизъюнктного кода) равна $|\mathcal{N}_n| = nN$. Если $\mathcal{A} = \mathcal{Q}_q$, то $|\mathcal{N}_n| = n\binom{q+1}{2}$.

Теорема 1. *Пусть $\mathfrak{K}$ — q-ичный разделяющий $(2,\, r)$-код и $\mathcal{A} = \{A_1,\ \ldots,\ A_q\}$, $|A_j| = m$, — $(2,\, q-2)$-семейство разделяющих множеств, содержащее q множеств. При любом q семейство $\mathcal{A}_n(\mathfrak{K})$ является $(2,\, r)$-семейством, непокрывающих множеств.*

Доказательство. Пусть $\boldsymbol{x},\ \boldsymbol{y} \in \mathfrak{K}$, $\boldsymbol{x} \neq \boldsymbol{y}$. Легко видеть, что

$$A_{\boldsymbol{x}} \cap A_{\boldsymbol{y}} = \left(A_{x_1}^{(1)} \cap A_{y_1}^{(1)}\right) \cup \ldots \cup \left(A_{x_n}^{(n)} \cap A_{y_n}^{(n)}\right). \quad (5)$$

Пусть $\{\boldsymbol{x}_1,\ \ldots,\ \boldsymbol{x}_r\} \subseteq \mathfrak{K} \setminus \{\boldsymbol{x},\ \boldsymbol{y}\}$. Код $\mathfrak{K}$ является разделяющим $(2,\, r)$-кодом. Поэтому существует такое s, что

$$(x_s \notin \{x_{s,1},\ \ldots,\ x_{s,r}\}) \ \&\ (y_s \notin \{x_{s,1},\ \ldots,\ x_{s,r}\}).$$

Очевидно, из последненго соотношения и (2) следует

$$\left(A_{x_s}^{(s)} \cap A_{y_s}^{(s)}\right) \not\subset \left(A_{x_{1,s}}^{(s)} \cup \ldots \cup A_{x_{r,s}}^{(s)}\right). \quad (6)$$

Поэтому $A_{\boldsymbol{x}} \cap A_{\boldsymbol{y}} \not\subset A_{\boldsymbol{x}_1} \cup \ldots \cup A_{\boldsymbol{x}_r}$. □

Эта теорема была впервые доказана в совместной работе О. Ю. Приходова и автора, которая была опубликована в 1969 г. в периодическом закрытом журнале, посвященным криптографической тематике.

Следует заметить, что теорема 1 не совпадает с похожей на нее теоремой, которая формулируется следующим образом (см. [3, Proposition 2.1], [4, Lemma 5], [7]):

Теорема 2. *Пусть $\mathfrak{K}$ является q-значным разделяющим $(2, r)$-кодом с параметрами $n \times |\mathfrak{K}|$ (n — длина $\mathfrak{K}$) и B — $(2, r)$-семейство, непокрывающих множеств (или, что одно и тоже, B — дизъюнктный $(2, r)$-код) размера $N_1 \times q$ (q — число подмножеств в B). Тогда каскадный код $\mathfrak{K} \diamond B$ является дизъюнктным $(2, r)$-кодом с параметрами $nN_1 \times |\mathfrak{K}|$.*

Замечание 6. Для того, чтобы теорема 2 была истинна, требуется, чтобы B был дизъюнктным $(2, r)$-кодом, а $\mathfrak{K}$ — разделяющим $(2, r)$-кодом. Теорема 1 показывает, что если в качестве B взять дизъюнктный $(2, q-2)$-код $\mathcal{A}$, при то это требование при $r > q-2$ является излишним. А именно, для построения дизъюнктного $(2, r)$-кода допускается использование любого q-значного разделяющего $(2, r)$-кода, где числа $q \geqslant 2$ и r могут быть взяты в любой комбинации.

Таким образом, построение дизъюнктного $(2, r)$-кода при любом r может быть сведена к построению q-ичного разделяющего $(2, r)$-кода, ибо для любого q существует q-ичный дизъюнктный $(2, q-2)$-код $\mathcal{Q}_q$.

В криптографии естественно использовать дизъюнктные $(2, r)$-коды, имеющие конкатенантную конструкцию, которые построены с помощью линейных разделяющих кодов.

3. Конструкции некоторых разделяющих и связанных с ними дизъюнктных кодов

3.1. Известная конструкция

3.1.1. Код Рида-Соломона как разделяющий код

Пусть $\mathbb{F}_q = \{\alpha_1, \ldots, \alpha_q\}$ и пусть $f = f(x) \in \mathbb{F}_q[x]$ и $\boldsymbol{a}_f = (f(\alpha_1), \ldots, f(\alpha_q))$ — вектор значений многочлена f. Код Рида-Соломона ($RS_{k,q}$-код), по определению, образован всеми векторами $\boldsymbol{a}_f$, у которых степень f не выше k, т. е.

$$RS_{k,q} = \{\boldsymbol{a}_f \mid \deg f \leqslant k\}. \tag{7}$$

Параметрами q-значного $RS_{k,q}$-кода являются (n, k, d), где $n = q$, $d = n - k + 1$ и q является примарным числом.

Лемма 2 (Сагалович [9, 3]). *Пусть $q > 2r$. Тогда*

$$RS_{\left[\frac{q}{2r}\right],\, q}, \qquad \left|RS_{\left[\frac{n}{2r}\right],\, n}\right| = q^{\left[\frac{n}{2r}\right]+1},$$

является разделяющим $(2, r)$-кодом.

Следствие 1 (из теоремы 1 и леммы 2). *Пусть $2r < q$. Тогда*

$$\mathcal{Q}_{q,q}\left(RS_{\left[\frac{q}{2r}\right],\, q}\right)$$

является разделяющим $(2, r)$-кодом при любом q.

Скоростью передачи дизъюнктивного (w, r)-кода $\mathfrak{K}$ длины N называется функция

$$R(\mathfrak{K}) = \frac{\log_2 |\mathfrak{K}|}{N}. \tag{8}$$

Автору неизвестен способ бесконечную последовательность дизъюнктивных $(2, r)$-кодов, где $r = \text{const} \geqslant 1$, $N \to \infty$, с ненулевой скоростью, используя только теорему 2 и лемму 2, но не используя теорему 1. Далее мы приведем один естественный рекуррентный способ построения бесконечной последовательности $\mathfrak{K}_1, \ldots, \mathfrak{K}_m, \ldots$, дизъюнктивных $(2, r)$-кодов без использования теоремы 1, которая имеет в пределе нулевую скорость, но скорость стремления к нулю очень медленная.

Вместе с тем построить бесконечную последовательность конкатенантных дизъюнктивных $(2, r)$-кодов, где $r = \text{const} \geqslant 1$, $N \to \infty$, с ненулевой скоростью, с помощью теоремы 1 достаточно просто. Это будет сделано в следующем разделе.

Указанная рекуррентная последовательность кодов с помощью леммы 2 строиться следующим образом.

Пусть $\mathfrak{K}_1$ — произвольный дизъюнктивный $(2, r)$-код длины N_1 и код $\mathfrak{K}_m$ уже построен. Пусть q_m — наибольшее примарное число такое, что $|\mathfrak{K}_m| = T_m \geqslant q_m$. Возьмем в качестве кода $\mathfrak{K}_{m+1}$ код

$$RS_{\left[\frac{q}{2r}\right],\, q} \diamond \mathfrak{K}_m,$$

который имеет длину $N_{m+1} = q_m N_m$ и число элементов $q^{[q_m/(2r)]+1}$. Покажем, что последовательность чисел $R(\mathfrak{K}_m)$ стремится к нулю.

Так как $R(\mathfrak{K}) \leqslant 1$, то, очевидно,

$$R(\mathfrak{K}_{m+1}) \sim \frac{\log_2 q_m}{2rN_m} \sim \frac{1}{2r} R(\mathfrak{K}_m), \qquad m \to \infty. \tag{9}$$

Отсюда при любом постоянном s следует, что

$$R(\mathfrak{K}_{m+1}) \lesssim \left(\frac{1}{2r}\right)^s, \tag{10}$$

то есть $\lim_{m\to\infty} R(\mathfrak{K}_m) = 0$.

Известна положительная оценка снизу величины

$$R(w,\, r) = \varlimsup_{N\to\infty} \frac{\log_2 T(N,\, w,\, r)}{N}, \tag{11}$$

где $T(N,\, w,\, r)$ — максимальное значение T при заданных N, w, r (см., например, [3, Section 3.5]). Из этой оценки следует, что существует бесконечная последовательность дизъюнктных $(2,\, r)$-кодов (не обязательно каскадных) с ненулевой скоростью.

3.2. Построение разделяющих $(w,\, 1)$-кодов

В настоящем разделе мы, используя теорему 1, получим необходимые и достаточные условия для существования разделяющих линейных $(w,\, 1)$-кодов.

Пусть $\boldsymbol{x},\, \boldsymbol{y} \in \mathbb{F}_q^n$. Произведением $\boldsymbol{x}\cdot\boldsymbol{y}$ векторов $\boldsymbol{x}$, $\boldsymbol{y}$ является вектор

$$\boldsymbol{x}\cdot\boldsymbol{y} = (x_1y_1,\, \ldots,\, x_ny_n). \tag{12}$$

Лемма 3. *Линейный над полем $\mathbb{F}_q$ код $\mathfrak{K}$ является разделяющим $(w,\, 1)$-кодом тогда и только тогда, когда для любых ненулевых векторов $\boldsymbol{x}_1,\, \ldots,\, \boldsymbol{x}_w \in \mathfrak{K}$ выполнено*

$$\boldsymbol{x}_1 \ldots \boldsymbol{x}_w \neq 0. \tag{13}$$

Доказательство. Пусть $\boldsymbol{x}'_1,\, \ldots,\, \boldsymbol{x}'_w,\, \boldsymbol{x} \in \mathfrak{K},\, \boldsymbol{x} \notin \{\boldsymbol{x}'_1,\, \ldots,\, \boldsymbol{x}'_w\}$.

Предположим, что для всех координат x_i у вектора $\boldsymbol{x}$ выполнено включение $x_i \in \{x'_{1,i},\, \ldots,\, x'_{w,i}\},\ i = 1,\, \ldots,\, n$. Тогда для векторов $\boldsymbol{x}_1 = \boldsymbol{x}'_1 - \boldsymbol{x},\, \ldots,\, \boldsymbol{x}_w = \boldsymbol{x}'_w - \boldsymbol{x}$, принадлежащих коду $\mathfrak{K}$, соотношение (13) не выполняется.

Наоборот, если существует координаты x_i у вектора $\boldsymbol{x}$, для которой $x_i \notin \{x'_{1,i},\, \ldots,\, x'_{w,i}\}$, тогда выполняется соотношение (13). □

Лемма 4. *Двоичный линейный код $\mathfrak{K}$ является разделяющим $(2,\, 1)$-кодом, если для любых $\boldsymbol{x},\, \boldsymbol{y} \in \mathfrak{K} \setminus \{0\}$*

$$\mathrm{wt}(\boldsymbol{x}+\boldsymbol{y}) < \mathrm{wt}(\boldsymbol{x}) + \mathrm{wt}(\boldsymbol{y}), \tag{14}$$

где $\mathrm{wt}(\boldsymbol{x})$ — вес вектора $\boldsymbol{x}$.

Доказательство. Если $\boldsymbol{x}\cdot\boldsymbol{y} = 0$, $\boldsymbol{x},\, \boldsymbol{y} \in \mathfrak{K} \setminus \{0\}$, то, очевидно,

$$\mathrm{wt}(\boldsymbol{x}+\boldsymbol{y}) = \mathrm{wt}(\boldsymbol{x}) + \mathrm{wt}(\boldsymbol{y}). \qquad \square$$

Следствие 2 ([9]). *Линейный над полем $\mathbb{F}_q$ код $\mathfrak{K}$ является разделяющим $(2,\, 1)$-кодом, если для любого $\boldsymbol{x} \in \mathfrak{K} \setminus \{0\}$ выполнено*

$$\frac{n}{3} < \mathrm{wt}(\boldsymbol{x}) < \frac{2n}{3}. \tag{15}$$

Доказательство. Если $\boldsymbol{x}\cdot\boldsymbol{y} = 0$, то $\mathrm{wt}(\boldsymbol{x}+\boldsymbol{y}) = \mathrm{wt}(\boldsymbol{x}) + \mathrm{wt}(\boldsymbol{y}) > 2n/3$, что противоречит предположению следствия. □

Следует сказать, что автору не известно конструктивных методов построения бесконечных семейств q-ичных разделяющих $(w,\, r)$-кодов при $q = \mathrm{const}$, $w > 1$, $n \to \infty$, которые имеют ненулевую скорость. Вместе с тем следствие 2, несмотря на свою простоту, позволяет доказать существование таких «хороших» кодов при $q = 2$.

А именно, хорошо известные методы получения границы Варшамова-Гилберта существования линейного кода с заданным кодовым расстоянием могут быть с помощью незначительных изменений трансформированы в методы получения границы существования для линейного кода, у которого ограничены снизу и свеху расстояния между парами элементов. При этом асимптотическое поведение границы не изменится.

Эта усовершенствованная граница позволяет доказать существование двоичных линейных кодов, для которых выполняется соотношение (15), с относительной скоростью $R = 1 - H_2(1/3) = 0.0817042$, что также доказывает существование линейных разделяющих $(2,\, 1)$-кодов.

Рассмотрим код, порождающая матрица которых является стандартной проверочной матрицей двоичного BCH-кода с удаленной единичной строкой и подходящим гарантированным кодовым расстоянием. Используя оценку А. Вейля, легко установить справедливость оценок (15) для таких кодов. К сожалению, этот способ позволяет построить конструктивным методом бесконечную последовательность раделяющих $(2,\, 1)$-кодов только с нулевой скоростью.

Заметим, что стандартными методами такими же как и для кодов, корректирующих ошибки, нетрудно получить нижние границы скорости, для которой существуют q-ичные разделяющие $(w,\, r)$-коды длины n (см., например, [1, Chapter 6]). Слово «скорость» в данном случае означает скорость кода в обычном теоретико-кодовым смысле. При $q,\, w,\, r = \mathrm{const}$, $n \to \infty$ эта скрость является положительной постоянной $R(q,\, w,\, r)$. Вместе с тем следует сказать, что эти оценки получены только кодов, которые, вообще говоря, не являются линейными.

Из теоремы 1 и теоремы 13 работы [1] (нижней границы существования двоичного разделяющего $(w,\, r)$-кода) непосредственно вытекает

Теорема 3. *Существует бесконечная последовательность каскадных дизъюнктных $(w,\, r)$-кодов для всех скоростей*

$$R(w,\, r) < \frac{1}{3} Y(w,\, r), \tag{16}$$

где

$$Y(w,\, r) = \frac{1}{w+r-1} \max_{0<p<1} \log_2\bigl((1 - p^w(1-p)^r - p^t(1-p)^w)^{-1}\bigr). \tag{17}$$

Следует сказать, что в работе [3] получены оценки $\underline{R}(w, r)$ снизу величины $R(w, r)$. В частности, $\underline{R}(2, 1) = 0.149$. В тоже самое время правая часть (16) для этого случая равна $0.2075/3 = 0.0691667$. Хотя это число заметно меньше числа $\underline{R}(2, 1)$, этот результат устанавливает положительность скорости линейного двоичного разделяющего $(2, 1)$-кода.

4. Схемы распределения ключей

Концепция схем распределения ключей включает следующие идеи (см. [2, 5, 7]).

1. Множество пользователей (в другой терминологии, множество вершин сети) абонентской сети связи идентифицируется с конечным множеством $\mathcal{Z}$, $|\mathcal{Z}| = T$. Обычно $\mathcal{Z}$ является множеством чисел или множеством векторов пространства $\mathbb{F}_q^n$. В рассматриваемых ниже случаях в качестве $\mathcal{Z}$ мы рассматриваем разделяющий код $\mathfrak{K} \in \mathbb{F}_q^n$.

2. Множество $\mathcal{L} = \{k_\alpha \mid \alpha \in \mathcal{N}\}$ является множеством всех ключей, использующихся в абонентской сети связи. Мы полагаем, что ключи индексируются элементами множества $\mathcal{N}$. Для простоты, вместо множества $\mathcal{L}$ принято рассматривать множество $\mathcal{N}$. Обычно, число элементов $\mathcal{N}$ обозначается как $N = |\mathcal{N}|$.

3. Пусть $\mathcal{L}_{\boldsymbol{z}} = \{k_{1,\,\boldsymbol{z}}, \ldots, k_{s_{\boldsymbol{z}},\,\boldsymbol{z}}\} \subset \mathcal{L}$ — множество ключей пользователя $\boldsymbol{z} \in \mathcal{Z}$, которое он хранит в своей памяти (электронной). Для простоты, вместо множества $\mathcal{L}_{\boldsymbol{z}}$ мы будем рассматривать множество $\mathcal{N}_{\boldsymbol{z}} = \{(1, \boldsymbol{z}), \ldots, (s_{\boldsymbol{z}}, \boldsymbol{z})\} \subset \mathcal{N}$. В рассматриваемых ниже случаях в качестве $\mathcal{N}_{\boldsymbol{z}}$ мы берем множество $A_{\boldsymbol{x}}$ (см. (4)).

4. Множество $\mathcal{L}_{\boldsymbol{z},\boldsymbol{z}'}$ общих ключей пары пользователей $\boldsymbol{z}, \boldsymbol{z}' \in \mathcal{Z}$ представляет собой множество $\mathcal{L}_{\boldsymbol{z}} \cap \mathcal{L}_{\boldsymbol{z}'}$. Вместо множества $\mathcal{L}_{\boldsymbol{z},\,\boldsymbol{z}'}$ мы будем рассматривать множество $\mathcal{N}_{\boldsymbol{z},\boldsymbol{z}'} = \mathcal{N}_{\boldsymbol{z}} \cap \mathcal{N}_{\boldsymbol{z}'}$. В упомянутом выше случае, в качестве $\mathcal{N}_{\boldsymbol{z},\,\boldsymbol{z}'}$ мы берем множество (см. (5))

$$\mathcal{N}_{\boldsymbol{z}} \cap \mathcal{N}_{\boldsymbol{z}'} = A_{\boldsymbol{x}} \cap A_{\boldsymbol{y}} = \bigcup_{j=1}^{n} (A_{x_j} \cap A_{y_j}). \tag{18}$$

5. Мы полагаем, что набор $\{\mathcal{N}_{\boldsymbol{z}} \mid \boldsymbol{z} \in \mathcal{Z}\}$ является $(2, r)$-семейством, разделяющих множеств, или в другой терминологии дизъюнктивным $(2, r)$-кодом.

Легко видеть, что в схеме распределения ключей, для которой выполнено это свойство, каждая коалиция $S \subset \mathcal{N}$, $|S| \leqslant r$, пользователей не может получить все общие ключи пары пользователей $\boldsymbol{z}, \boldsymbol{z}' \notin S$, т. е. всегда у пары $\boldsymbol{z}, \boldsymbol{z}'$ имеется, по меньшей мере один общий ключ из $\mathcal{L}_{\boldsymbol{z},\,\boldsymbol{z}'}$, который не входит в объединение ключей недобросовестных пользователей из коалиции S. Такие схемы называются схемами распределения ключей, устойчивыми к r компрометациям.

6. Сложностью семейства, разделяющих множеств $\mathcal{N}$, или соответствующего дизъюнктного кода естественно называть число

$$a(\mathcal{N}) = \max_{\boldsymbol{z} \in \mathcal{Z}} |\mathcal{N}_{\boldsymbol{z}}| \tag{19}$$

и минимизировать его при заданном числе T. Заметим, что этот параметр не совпадает с общепринятым параметром N — длиной дизъюнктного кода, который минимизируется во всех известных автору работах. Например, сложность (в нашей терминологии) тривиального $(2, q-2)$-семейства Q_q равна $a(Q_q) = q$, а сложность каскадного семейства $\mathcal{Q}_{q,n}(\mathfrak{K})$ равна $a(\mathcal{Q}_{q,n}(\mathfrak{K})) = qn$.

Список литературы

[1] *Cohen G. D.*, *Schaathum H. G.* Asymptotic Overview on Separating Codes. Report No 248, May 2003, Bergen, Norway.

[2] *Mitchell Ch. J.*, *Pipper F. C.* Key storage in secure network. Discrete Applied Mathematics, 21, 215–228, 1988.

[3] *D'yachkov A.*, *Vilenkin P.*, *Macula A.*, *Torney D.* Families of finite sets in which no intersectin of l sets is covered by the union of s other. J. of Combinatorial Theory, S. A 99, 195–218, 2002.

[4] *Hyun Kwang Kim*, *Lebedev V.* On optimal superimposed codes.

[5] *Quinn K. A. S.* Some constructions for key distribution pattens. Designs, Codes and Cryptography, 4, 177–191, 1994.

[6] *Quinn K. A. S.* Bounds for key distribution pattens. J. Cryptology, 12, 227–239, 1999.

[7] *Stinson D. R.* On some methods for unconditionally secure key distribution and broadcast encryption. Designs, Codes and Cryptography, 12, 215–243, 1997.

[8] *MacWilliams F. W.*, *Sloane N. W. A.* The Theory of Error-Correcting Codes. North-Holland, Amsterdam, 1977. (Русский перевод: Ф. Дж. МакВильямс, Н. Дж. А. Слоэн. Теория кодов, исправляющих ошибки. М.: Связь, 1979.)

[9] *Sagalovich Yu. L.* On separating systems. Problemy Peredachi Informatsii, 30, 14–35, 1994 (in Russian).

[10] *Лебедев В. С.* Асимптотическая верхняя оценка граница для скорости кодов, свободных от (w, r)-перекрытий. Проблемы передачи информации, т. 39, № 4, 3–10, 2003.

[11] *Ким Ш. Х.*, *Лебедев В. С.* Об оптимальности тривиальных кодов, свободных от (w, r)-перекрытий. Проблемы передачи информации, т. 40, № 3, 13–20, 2004.

О значениях аффинного ранга носителя спектра платовидных функций

Ю. В. Таранников

Платовидные функции представляют большой интерес в криптографии для изучения бент функций и в силу того, что многие криптографически важные функции являются платовидными. В этой работе изучаются возможные значения аффинного ранга носителя спектра платовидных функций. Мы доказываем, что аффинный ранг любой платовидной функции с носителем спектра мощности 16 равен 4, 5 или 6. Мы рассматриваем для любого натурального h платовидные функции с носителем спектра мощности 4^h, даем оценки аффинного ранга для таких функций и строим функции, аффинный ранг которых принимает все возможные значения от $2h$ до $2^{h+1}-2$.

Мы рассматриваем $\mathbf{F}_2^n$, векторное пространство наборов длины n с компонентами из $\mathbf{F}_2$ — конечного поля из двух элементов 0 и 1, операции сложения и умножения в котором вводятся как обычные операции сложения и умножения чисел 0 и 1 по модулю 2.

Весом $\mathrm{wt}(x)$ набора x из $\mathbf{F}_2^n$ называется число единиц в x. *Вес* $\mathrm{wt}(f)$ функции f над $\mathbf{F}_2^n$ — это число наборов x из $\mathbf{F}_2^n$, для которых $f(x)=1$. Функция f называется *уравновешенной*, если $\mathrm{wt}(f)=\mathrm{wt}(f\oplus 1)=2^{n-1}$ (т. е. функция принимает значения 0 и 1 на одинаковом числе наборов). Наборы x' и x'' называются *соседними*, если они различаются только в i-й компоненте. Обозначим через x^i набор, который отличается от x только в i-й компоненте, $i=1,\ldots,n$. Переменная x_i называется *фиктивной* для функции f, если для любых наборов x' и x'', соседних по i-й компоненте, выполнено $f(x')=f(x'')$. *Расстоянием Хэмминга* $d(x', x'')$ между двумя наборами x' и x'' называют число компонент, в которых наборы x' и x'' различаются. Для заданной функции f из $\mathbf{F}_2^n$ минимум расстояний $d(f, l)$, где l пробегает множество всех аффинных функций из $\mathbf{F}_2^n$ называется *нелинейностью* функции f и обозначается через $\mathrm{nl}(f)$. *Подфункцией* булевой функции f называется функция f', полученная подстановкой в f некоторых констант 0 или 1 вместо некоторых переменных.

Пусть $x=(x_1,\ldots,x_n)$ и $u=(u_1,\ldots,u_n)$ — это наборы длины n над $\mathbf{F}_2$. *Скалярное произведение* x и u — это функция, которая определяется как

$$\langle x,\, u\rangle=\sum_{i=1}^{n} x_i u_i,$$

где сложение и умножение берутся над $\mathbf{F}_2$. Под суммой $x+u$ двух двоичных наборов x и u мы понимаем их покомпонентное сложение над $\mathbf{F}_2$.

Преобразованием Уолша булевой функции f называется целочисленная функция над $\mathbf{F}_2^n$, определяемая следующим образом

$$W_f(u)=\sum_{x\in\mathbf{F}_2^n}(-1)^{f(x)+\langle u,x\rangle}.$$

Для каждого $u\in\mathbf{F}_2^n$ значение $W_f(u)$ называется *коэффициентом Уолша*. Коэффициенты Уолша будем называть *спектральными коэффициентами*, а совокупность всех 2^n коэффициентов Уолша — *спектром* булевой функции. Коэффициенты Уолша удовлетворяют формуле обращения $(-1)^{f(x)}=2^{-n}\sum_{u\in\mathbf{F}_2^n}W_f(u)(-1)^{\langle u,x\rangle}$ и равенству Парсеваля $\sum_{u\in\mathbf{F}_2^n}W_f^2(u)=2^{2n}$. Нелинейность булевой функции f выражается через ее коэффициенты Уолша следующим образом:

$$\mathrm{nl}(f)=2^{n-1}-\frac{1}{2}\max_{u\in\mathbf{F}_2^n}|W_f(u)|.$$

Множество S_f всех наборов u, таких что $W_f(u)\neq 0$, называется *носителем спектра* функции f.

Булева функция называется *бент-функцией*, если значение коэффициентов на всех наборах равно $\pm 2^{n/2}$. Бент-функции существуют при всех четных n, а при нечетных — не существуют. Бент-функция является функцией с максимально возможной нелинейностью $2^{n-1}-2^{(n/2)-1}$ среди всех функций от n переменных при четном n. Булева функция называется *платовидной*, если ее коэффициенты Уолша принимают ровно три возможных значения: 0 и $\pm 2^c$ для некоторого c. Платовидные функции представляют большой интерес для изучения бент-функций (например, потому, что при разложении бент-функции по переменной возникают две платовидные функции), а также потому, что многие криптографически важные функции являются платовидными (например, m-устойчивые функции с максимально возможной для них нелинейностью $2^{n-1}-2^{m+1}$). Обозначим для платовидных функций $\varphi(x)=2^{-c}W_f(x)$. Тогда для любого $x\in\mathbf{F}_2^n$ величина $\varphi(x)$ может принимать только три значения: 0, -1 и 1. Множество S_f всех наборов u, таких

что $W_f(u) \neq 0$, называется *носителем спектра* платовидной функции. Множество всех наборов, на которых $\varphi(x) = -1$, будем обозначать через T^-, а множество всех наборов, на которых $\varphi(x) = 1$, будем обозначать через T^+. Из равенства Парсеваля сразу следует, что мощность носителя спектра равна 4^{n-c}. Бент-функцию удобно рассматривать как частный случай платовидной при $c = n/2$ и $|S_f| = 2^n$, что мы и будем часто делать с оговорками. (Хотя часто формально бент-функции к платовидным не относят.)

Платовидные функции изучались в большом числе работ, см., например, [1], [4], [8].

Для каждого $u \in \mathbf{F}_2^n$ *автокорреляционный коэффициент* функции f на наборе u определяется как $\Delta_f(u) = \sum_{x \in \mathbf{F}_2^n} (-1)^{f(x)+f(x+u)}$. Функция $D_u f = f(x) + f(x+u)$ называется *производной* функции f по направлению u. Множество наборов $u \in \mathbf{F}_2^n$, таких что $D_u \equiv \mathrm{const}$, называется множеством *линейных структур* функции f. Легко проверить, что множество линейных структур функции f образует линейное пространство в $\mathbf{F}_2^n$. Наличие у функции нетривиальной линейной структуры в некоторых случаях (но не всех) является криптографической слабостью.

Пусть E — произвольное подмножество $\mathbf{F}_2^n$. *Рангом* множества E называется называется размерность подпространства, порожденного E в $\mathbf{F}_2^n$. *Аффинным рангом* множества E называется называется размерность наименьшего класса смежности в $\mathbf{F}_2^n$, содержащего E. Ранг и аффинный ранг носителя спектра булевой функции будем обозначать через k и $\mathbf{k}$, соответственно. Для краткости в данной работе аффинный ранг и ранг носителя спектра булевой функции мы будем называть просто ее *аффинным рангом* и *рангом*, соответственно. Легко убедиться, что $\mathbf{k} \in \{k, k-1\}$. Известно (см., например, [5]), что размерность линейной структуры функции f равна $n - \mathbf{k}$. Если существует набор $u \in \mathbf{F}_2^n$, такой что $D_u \equiv 1$, то $k = \mathbf{k} + 1$. Если такого набора не существует, то $k = \mathbf{k}$.

Дополнительные сведения о свойствах булевых функций можно найти в [2] и [3].

Спектры функций f и f', переводимых один в другой аффинным преобразованием спектра, называются *аффинно эквивалентными*. Спектры функций f и f', переводимых один в другой линейным преобразованием спектра, называются *линейно эквивалентными*. Аналогично, из аффинной эквивалентности функций не следует аффинная эквивалентность их спектров. Например, потому, что при аффинном преобразовании функции f величина $\mathrm{wt}(f)$ остается неизменной, но

$\mathrm{wt}(f) = 2^{n-1} - \frac{1}{2} W_f(0)$, поэтому переводя при аффинном преобразовании спектра в $\overline{0}$ набор с другим значением коэффициента Уолша, мы получим функцию, не являющуюся аффинно эквивалентной f. В то же время линейное преобразование спектра является линейным преобразованием функции, и наоборот.

Очевидно, аффинное преобразование спектра платовидной функции f переводит его в спектр также платовидной некоторой функции f' с той же мощностью носителя спектра, а аффинное преобразование платовидной функции f переводит ее в платовидную функцию f' с той же мощностью носителя спектра.

Изучение платовидных функций на $\mathbf{F}_2^n$ с носителем спектра мощности 4^h можно в некотором смысле свести к изучению платовидных функций с носителем спектра той же мощности 4^h, заданных на $\mathbf{F}_2^{\mathbf{k}}$. Более того, если $\mathbf{k} > 2h$, то любую платовидную функцию f' на $\mathbf{F}_2^n$ с носителем мощности 4^h можно получить из некоторой функции f на $\mathbf{F}_2^{\mathbf{k}}$ с носителем той же мощности 4^h, добавив $n - \mathbf{k}$ фиктивных переменных и выполнив некоторое линейное преобразование функции. Того же можно добиться и в случае, если $\mathbf{k} = 2h$ и $W_{f'}(0) \neq 0$ (в этом случае функция f будет бент-функцией). Если $\mathbf{k} = 2h$ и $W_{f'}(0) = 0$, то указанного линейного преобразования функции не существует, но можно использовать аффинное преобразование спектра, либо же взять функцию f от $\mathbf{k} + 1$ переменной.

Заметим, что указанного сведения может быть недостаточно, если требуется исследовать дополнительные свойства функций, не сохраняющиеся при аффинных преобразованиях (например, корреляционную иммунность).

Укажем также на следующее свойство, позволяющее не рассматривать специально вопрос о возможных значениях, принимаемых рангом k.

Лемма. *Если существует платовидная функция с носителем спектра мощности 4^h и аффинным рангом, равным $\mathbf{k}$, то существуют платовидные функции с носителем спектра той же мощности 4^h и рангами, равными и $k = \mathbf{k}$, и $k = \mathbf{k} + 1$.*

Из вышесказанного следует, что аффинный ранг является важной характеристикой платовидных функций. Очевидно, что аффинный ранг любой платовидной функции с носителем спектра мощности 4^h не меньше $2h$, потому что меньшие классы смежности не содержат 4^h наборов. Любой платовидная функция с носителем спектра мощности 1 есть аффинная функция и, очевидно, ее аффинный ранг равен 0. Аффинный ранг любой платовидной функции с носителем спектра мощности 4 ра-

вен 2. Этот факт доказан в [7], но, по-видимому, был известен намного раньше. Платовидные функции с носителем спектра мощности 16 (не называясь платовидными) фактически рассматривались в [6], а в работе [5] для подкласса платовидных функций с носителем спектра мощности 16 (более точно, для кубических устойчивых порядка $n-4$ функций) была получена оценка $\mathbf{k} \leqslant k \leqslant 9$. Автором доказано, что аффинный ранг любой платовидной функции с носителем спектра мощности 16 равен 4, 5 или 6. Кроме того, рассмотрены для любого натурального h платовидные функции с носителем спектра мощности 4^h, даны оценки аффинного ранга для таких функций и построены функции, аффинный ранг которых принимает все возможные значения от $2h$ до $2^{h+1}-2$.

Теорема 1. *Пусть f является платовидной функцией, $|S_f| = 16$. Тогда для аффинного ранга* $\mathbf{k}$ *носителя спектра S_f справедливо неравенство* $\mathbf{k} \leqslant 6$.

Теорема 2. *Для любого натурального s, удовлетворяющего неравенствам $2h \leqslant s \leqslant 2^{h+1}-2$ существует платовидная функция с носителем спектра мощности 4^h и аффинным рангом s.*

Следствие. *Аффинный ранг платовидной функции с носителем спектра мощности* 16 *может принимать только значения* 4, 5 *и* 6.

Теорема 3. *Пусть f является платовидной функцией, $|S_f| = 4^h$. Тогда для аффинного ранга* $\mathbf{k}$ *носителя спектра S_f справедливо неравенство* $\mathbf{k} \leqslant 2^{2h-1} - 2^{h-1} + h$.

При доказательстве теорем использовались комбинаторно-алгебраическая техника и оригинальные конструкции.

При $h = 2$ оценка теоремы 3 не достигается.

Гипотеза. Для любого натурального h максимально возможный аффинный ранг платовидной функции с носителем спектра мощности 4^h равен $2^{h+1}-2$.

Список литературы

[1] *Кузнецов Ю. В.* О носителях платовидных функций. Материалы VIII Международного семинара «Дискретная математика и ее приложения» (2–6 февраля 2004 г.). М.: Изд-во механико-математического факультета МГУ, 2004. С. 424–426.

[2] *Логачев О. А.*, *Сальников А. А.*, *Ященко В. В.* Булевы функции в теории кодирования и криптографии. М.: Изд-во МЦНМО, 2004.

[3] *Тарannиков Ю. В.* О корреляционно-иммунных и устойчивых булевых функциях. Математические вопросы кибернетики. Вып. 11. М.: Физматлит, 2002. С. 91–148.

[4] *Тарannиков Ю. В.* О платовидных устойчивых функциях. Материалы VIII Международного семинара «Дискретная математика и ее приложения» (2–6 февраля 2004 г.). М.: Изд-во механико-математического факультета МГУ, 2004. С. 431–435.

[5] *Carlet C.*, *Charpin P.* Cubic Boolean functions with highest resiliency. Proceedings of 2004 IEEE International Symposium on Information Theory (ISIT 2004). Chicago, USA, June 2004. P. 497. Full version is submitted to IEEE Transactions on Information Theory.

[6] *Kasami T.*, *Tokura N.*, *Azumi S.* On the weight enumeration of weights less than $2.5d$ of Reed–Muller codes. Information and Control. Vol. 30(4). April 1976. P. 380–395.

[7] *Pei D.*, *Qin W.* The correlation of a Boolean function with its variables. Progress in Cryptology — Indocrypt 2000. Lecture Notes in Computer Science. V. 1977. Springer-Verlag, 2000. P. 1–8.

[8] *Zheng Y.*, *Zhang X.-M.* Plateaued functions. Proceedings of ICICS 1999. Lecture Notes in Computer Science. V. 1726. Springer-Verlag, 1999. P. 284–300.

О распознавании признаков элементов в конечных группах

В. М. Фомичёв

Предложены общие подходы к постановке и решению задачи дифференциации элементов конечной группы, порожденной при помощи системы S образующих. В качестве меры сложности порождения подмножества H группы рассматривается кратчайшая из длин слов в алфавите S, представляющих элементы подмножества H.

1. Основные понятия и исследовательские задачи, возникающие при изучении признаков в конечных группах

Пусть Φ — конечная группа, $\langle S\rangle$ — группа, порожденная системой образующих $S=\{s_1, s_2, \ldots, s_p\}$, где $S\subseteq\Phi$ и $p=|S|>0$; $G<\langle S\rangle$. Обозначим через $L(g, S)$ длину элемента g группы G в системе образующих S [1].

Определение 1. Если $\langle S\rangle\cap H\neq\varnothing$, то показателем множества H в системе образующих S (обозначается $\mathrm{pok}_S H$) назовем наименьшую из длин всех элементов множества H в системе образующих S, то есть $\mathrm{pok}_S H=\min_{g\in G} L(g, S)$. Если $\langle S\rangle\cap H=\varnothing$, то положим $\mathrm{pok}_S H=\infty$.

Определение 2. Группа G имеет H-признак, если $G\cap H\neq\varnothing$, при этом H-признак тривиален (нетривиален), если $G\cap H$ — одноэлементное множество (если $G\cap H$ содержит более одного элемента). Группа G не имеет H-признака, если $G\cap H=\varnothing$.

Определение 3. Если группа G имеет H-признак, то показателем H-признака группы G (или H-показателем группы G) в системе образующих S назовем показатель множества $G\cap H$ в системе образующих S, то есть $\mathrm{pok}_S(G\cap H)$. Если $G=\langle S\rangle$, то $\mathrm{pok}_S(G\cap H)=\mathrm{pok}_S H$.

В связи с данными определениями представляет интерес, в частности, для криптологии, ряд задач исследования строения группы $\langle S\rangle$:

1. Распознавание наличия H-признака группы G.
2. Описание множества $G\cap H$ и определение (или оценка) H-показателя группы G в системе образующих S.

Если $G\cap H=\{g\}$ (H-признак группы G тривиален), то последняя задача сводится к задаче определения или оценки величины $L(g, S)$.

3. Для заданных G, H исследование зависимости величины $\mathrm{pok}_S(G\cap H)$ от выбора системы S образующих элементов группы G.

Многообразие указанных задач определяется многообразием выбора групп G, множеств S образующих элементов и классов признаков H.

2. Оценки показателей групповых признаков в конечных группах

Определение 4. Если $G\cap H<G$ то H-признак в группе G назовем групповым признаком.

Определение 5. Наименьшим разбросом множества $H\subseteq V$ в графе $\Gamma(V, U)$ с множеством вершин V и множеством дуг U назовем число $\sigma_\Gamma(H)=\min_{i,j\in H}\rho_\Gamma(i, j)$, где $\rho_\Gamma(i, j)$ — длина кратчайшего пути из вершины i в вершину j в графе $\Gamma(V, U)$.

Теорема 1. *Если группа $\langle S\rangle$ имеет групповой H-признак, то*

$$\mathrm{pok}_S H=\sigma_\Gamma(\langle S\rangle\cap H)\leqslant|\langle S\rangle:(\langle S\rangle\cap H)|,$$

где Γ_S — граф Кэли группы $\langle S\rangle$, построенный по системе образующих S [2, 3], *и $|\langle S\rangle:(\langle S\rangle\cap H)|$ — индекс подгруппы $\langle S\rangle\cap H$ в группе $\langle S\rangle$.*

Верхняя оценка теоремы 1 достигается для циклических групп.

Теорема 2. *Если группа $\langle g\rangle$ имеет групповой H-признак, то* $\mathrm{pok}_g H=|\langle g\rangle:(\langle g\rangle\cap H)|$.

Нижняя оценка величины $\mathrm{pok}_S H$ существенно зависит от свойств группы $\langle S\rangle\cap H$ и системы S образующих элементов и может достигать малых значений. Например, $\mathrm{pok}_S H=1$, если $S\cap H\neq\varnothing$, и $\mathrm{pok}_S H=2$, если $S\cap H=\varnothing$ и система S содержит инволюции или взаимно обратные элементы.

3. Свойства монотонных признаков в конечных группах

Элементы в циклических группах «наследуют» некоторые свойства порождающего элемента, т. е. если элемент g обладает некоторым свойством, то и любой другой элемент группы $\langle g\rangle$ обладает этим свойством.

Определение 6. Пусть $G\cap H\neq\varnothing$, тогда H-признак группы G назовем монотонным, если из включения $g\in G\cap H$ следует, что $\langle g\rangle\subseteq G\cap H$.

Изучение строения монотонного H-признака группы G (то есть множества $G\cap H$) можно свести к изучению строения монотонного H-при-

знака циклических подгрупп группы G. Рассмотрим циклическую группу $\langle g\rangle$ порядка n.

Лемма 1. *Если группа $\langle g\rangle$ имеет монотонный H-признак, то $g^t \in H$, где $t \in \{1, \ldots, n\}$, тогда и только тогда, когда $g^d \in H$, где $d = (t, n)$.*

Таким образом, состав множества $\langle g\rangle \cap H$ определяется выполнимостью включений $g^d \in H$ для чисел d из решетки $D(n)$ делителей натурального числа n.

Определение 7. Пусть $n_1, \ldots, n_m$ — натуральные числа, $m \geqslant 1$. Число n_i, где $i \in \{1, \ldots, m\}$, назовем простым во множестве $M = \{n_1, \ldots, n_m\}$ (или M-простым), если n_i не делится ни на одно из чисел множества M, отличных от n_i. Множество всех M-простых чисел, обозначим $\mathrm{prm}\ M$.

Определение 8. Если группа $\langle g\rangle$ имеет монотонный H-признак, то $(H,\ g)$-пороговым числом назовем всякое число, простое во множестве $\{d \in D(n) : g^d \in H\}$. Множество всех $(H,\ g)$-пороговых чисел обозначим $\Pi(H,\ g)$.

Значит, $(H,\ g)$-пороговое число есть делитель d числа n такой, что $g^d \in H$ и $g^\tau \notin H$ для любого собственного делителя τ числа d. Множество $\Pi(H,\ g)$ образует антицепь в решетке $D(n)$ и имеет вид:

$$\Pi(H,\ g) = \mathrm{prm}\{d \in D(n) :\ g^d \in H\}.$$

Теорема 3. *В группе $\langle g\rangle$, имеющей монотонный H-признак, $g^t \in H$ в том и только в том случае, если t кратно хотя бы одному из $(H,\ g)$-пороговых чисел, то есть*

$$\langle g\rangle \cap H = \bigcup_{t \in \Pi(H, g)} \langle g^t\rangle.$$

H-признак в группе $\langle g\rangle$ является групповым тогда и только тогда, когда $|\Pi(H,\ g)| = 1$.

Следствие. *Пусть группа $\langle g\rangle$ имеет монотонный H-признак и $\Pi(H,\ g) = \{t_1, \ldots, t_r\}$. Тогда $\mathrm{pok}_g\ H = \min\{t_1, \ldots, t_r\}$.*

4. Взаимосвязь монотонных признаков в конечных группах и заданных на группах монотонных функций

Рассмотрим функцию f, определенную на конечной группе G и принимающую значения в линейно или частично упорядоченном множестве Y, где $|Y| > 1$.

Определение 9. Ограничение функции f на циклическую подгруппу $\langle g\rangle$ группы G назовем g-подфункцией функции f, $g \in G$.

Любой элемент группы $\langle g\rangle$ имеет вид g^t, где $t \in \{1, \ldots, n\}$, поэтому g-подфункцию можно задать как функцию $f_g(t)$: $\{1, \ldots, n\} \to Y$, где $f_g(t) = f(g^t)$.

Определение 10. Назовем g-подфункцию $f_g(t)$ монотонной (антимонотонной) на решетке $D(n)$, если для любых τ, $t \in \{1, \ldots, n\}$ таких, что (τ, n) делит (t, n), выполнено неравенство $f(\tau) \leqslant f(t)$ $(f(\tau) \geqslant f(t))$.

Определение 11. Функцию f назовем монотонной (антимонотонной) на группе G, если при любом $g \in G$ ее g-подфункция $f_g(t)$ монотонна (антимонотонна) на решетке $D(\mathrm{ord}\ g)$.

Монотонной (антимонотонной) на группе G является, например, функция $f(g) = (\mathrm{ord}\ g)^{-1}$ (функция $f(g) = \mathrm{ord}\ g$).

Пусть $f\colon G \to Y$, и функция $f(g)$ сюръективна. Множество всех элементов g группы G, удовлетворяющих при $y \in Y$ условию $f(g) \geqslant y$ $(f(g) \leqslant y)$, обозначим $G(f \geqslant y)$ $(G(f \leqslant y))$.

Теорема 4. *Если функция f монотонна (антимонотонна) на группе G, то при любом $y \in Y$ группа G имеет монотонный $G(f \geqslant y)$-признак $(G(f \leqslant y)$-признак).*

Если группа G имеет монотонный H-признак, то существует монотонная (антимонотонная) на группе G функция f такая, что $G \cap H = G(f \geqslant y)$ $(G \cap H = G(f \leqslant y))$ при некотором $y \in Y$.

Список литературы

[1] *Глухов М. М.* О числовых параметрах, связанных с заданием конечных групп системами образующих элементов. В сб. «Труды по дискретной математике», т. 1, М.: ТВП, 1997, с. 43–66.

[2] *Гроссман И.*, *Магнус В.* Группы и их графы. М.: Мир, 1971, 248 с.

[3] *Магнус В.*, *Каррас А.*, *Солитэр Д.* Комбинаторная теория групп. М.: Наука, 1974, 456 с.

Распознавание признака унидоминантности в группе подстановок

В. М. Фомичёв

В соответствии с данным в [1] общим подходом к постановке и решению задачи дифференциации элементов конечной группы рассмотрены вопросы дифференциации элементов группы подстановок G по определенному структурному признаку. Результаты могут быть использованы, в частности, для определения линейной подгруппы группы подстановок векторного пространства над конечным полем.

Пусть $\Phi(X)$ — группа всех подстановок конечного множества X, $g \in \Phi(X)$. Положим, что $\operatorname{ord} g = n$ и цикловая структура подстановки g состоит из k_i циклов длины l_i, $i = 1, \ldots, m$. Обозначим $L(g)$ множество всех различных длин циклов подстановки g, $L(g) = \{l_1, \ldots, l_m\}$.

Определение 1. Число l_i назовем доминирующим во множестве $L(g)$, $i \in \{1, \ldots, m\}$, если l_i не делит никакого другого числа из множества $L(g)$. Множество всех чисел, доминирующих во множестве $L(g)$, назовем доминантой множества $L(g)$ и обозначим $\operatorname{dom} L(g)$. Для определенности положим: $\operatorname{dom} L(g) = \{l_1, \ldots, l_d\}$, где $1 \leqslant d \leqslant m$.

Определение 2. Подстановку g назовем d-доминантной, если $|\operatorname{dom} L(g)| = d > 1$, и унидоминантной, если $|\operatorname{dom} L(g)| = 1$. Множество всех унидоминантных подстановок множества X обозначим U.

Исследуем свойства U-признака в группе подстановок G, где $G < \Phi(X)$. Группа подстановок G имеет U-признак, так как $L(e) = (1)$ и, следовательно, $e \in G \cap U$.

Пусть $f(g) = |\operatorname{dom} L(g)|$, то есть f — функция, определенная на множестве подстановок $\Phi(X)$ и принимающая натуральные значения.

Утверждение 1. *Функция f антимонотонна на группе $\Phi(X)$, поэтому всякая подгруппа G группы $\Phi(X)$ имеет монотонный U-признак.*

Обозначим через M^* множество всех попарно различных чисел набора M натуральных чисел.

Связь между цикловой структурой подстановки g и множеством (U, g)-пороговых чисел, определяющих строение множества $\langle g \rangle \cap U$, описывается теоремой 1.

Теорема 1. *Пусть* $\operatorname{dom} L(g) = \{l_1, \ldots, l_d\}$, *где* $d > 1$, *и верны канонические разложения чисел*:

$$n = p_1^{k_1} \ldots p_s^{k_s}, \qquad l_i = p_1^{k_{i1}} \ldots p_s^{k_{is}},$$

где $p_1, \ldots, p_s$ — *простые числа*, $k_1, \ldots, k_s$ — *натуральные числа*, $k_{i1}, \ldots, k_{is}$ — *целые неотрицательные числа*, $i = 1, \ldots, d$, *и* $k_j = \max\{k_{1j}, \ldots, k_{dj}\}$ *для* $j = 1, \ldots, s$. *Тогда*

$$\Pi(U, g) = \operatorname{prm}(\{u_1, \ldots, u_d\}^*),$$

где для $i = 1, \ldots, d$

$$u_i = p_1^{w_{i1}k_1} \ldots p_h^{w_{ih}k_h}$$

и для $j = 1, \ldots, h$

$$w_{ij} = \begin{cases} 1, & k_{ij} < k_j; \\ 0, & k_{ij} = k_j. \end{cases}$$

Следствие 1. *Нетривиальная группа подстановок G имеет нетривиальный монотонный U-признак, который не является групповым.*

Следствие 2. $\langle g \rangle \cap U = \bigcup_{t \in \operatorname{prm}(\{u_1, \ldots, u_d\}^*)} \langle g^t \rangle$, *вследствие этого* $\operatorname{pok}_g U = \min\{u_1, \ldots, u_d\}$.

Для группы подстановок G, порожденной системой образующих S, где $S = \{s_1, \ldots, s_p\}$, верна оценка:

$$\operatorname{pok}_S U \leqslant \min\{\Pi(U, s_1) \cup \ldots \cup \Pi(U, s_p)\}.$$

Обозначим:

P^r — векторное пространство размерности r над полем P порядка q,

$\mathrm{GL}(r, q)$ — полная линейная группа преобразований пространства P^r,

θ — нулевой вектор пространства P^r,

G — группа подстановок пространства P^r,

G_θ — стабилизатор элемента θ в группе G,

U_G — множество всех унидоминантных подстановок группы G.

Теорема 2. *Группа G имеет групповой* $\mathrm{GL}(r, q)$*-признак, где* $G \cap \mathrm{GL}(r, q) \subseteq G_\theta \cap U_G$.

Список литературы

[1] *Фомичёв В. М.* О распознавании признаков элементов в конечных группах. МаБИТ-04.

Дискретные временные ряды с «длинной памятью» и их использование в задачах защиты информации

Ю. С. Харин, А. Н. Ярмола

1. Введение

Статистическое тестирование дискретных случайных и псевдослучайных последовательностей $x_t \in \mathcal{A} = \{0, 1, \ldots, N-1\}$, $t \in \mathbb{N}$ — одна из важнейших проблем криптографической защиты информации [1]. Статистический тест — это решающее правило, позволяющее по наблюдаемой реализации $x_1, \ldots, x_n \in \mathcal{A}$ длительностью n с заданной точностью принять гипотезу H_0 ($\{x_t\}$ — равномерно распределенная случайная последовательность (РРСП), т. е. символы $x_1, x_2, \ldots$ независимы в совокупности и равномерно распределены на $\mathcal{A}$) или принять альтернативу H_1. Проведенный в [2] обзор существующих статистических тестов показывает: 1) многие из известных тестов ориентированы на проверку лишь одного из вероятностных свойств, характеризующих РРСП; 2) многие тесты построены «эвристически» и не фиксируют семейство альтернатив; 3) многие тесты не имеют оценок мощности. Поэтому актуальны задачи разработки адекватных вероятностных моделей для описания отклонений H_1 от модели РРСП, построения алгоритмов статистического анализа для обнаружения и оценивания таких отклонений.

Настоящий доклад посвящен решению этих задач применительно к отклонениям H_1 от модели РРСП, характеризующимся наличием стохастических зависимостей высокого порядка в $\{x_t\}$. Для этой цели предлагается использовать «малопараметрические» модели «временных рядов с длинной памятью» («long-memory time series») [3, 4, 5, 6].

2. Использование цепи Маркова s-го порядка с r частичными связями ЦМ(s, r)

Пусть $(\Omega, \mathcal{F}, \mathsf{P})$ — вероятностное пространство; $x_t \in \mathcal{A} = \{0, 1, \ldots, N-1\}$, $t \in \mathbb{N}$ — однородная цепь Маркова s-го порядка ($s, N \in \mathbb{N}$) с $(s+1)$-мерной матрицей вероятностей переходов $P = (p_{i_1, \ldots, i_s, i_{s+1}})$,

$$p_{i_1, \ldots, i_s, i_{s+1}} = \mathsf{P}\{x_t = i_{s+1} \mid x_{t-1} = i_s, \ldots, x_{t-s} = i_1\}, \quad i_1, \ldots, i_{s+1} \in \mathcal{A}.$$

Для такой модели число параметров растет экспоненциально быстро при увеличении порядка s, поэтому актуальна проблема построения «малопараметрических» моделей цепей Маркова высокого порядка. В [3] предложена MTD-модель, использующая вероятностные смеси простых цепей Маркова. В данном докладе исследуется модель ЦМ(s, r) цепи Маркова с r частичными связями, предложенная в [5]:

$$p_{i_1, \ldots, i_s, i_{s+1}} = q^0_{i_{m_1^0}, \ldots, i_{m_r^0}, i_{s+1}}, \quad i_1, \ldots, i_{s+1} \in \mathcal{A}, \tag{1}$$

параметрами которой являются: $r \in \{1, \ldots, s\}$ — число связей;

$$M_r^0 = \left(m_1^0, \ldots, m_r^0\right) \in \mathcal{M}$$

— произвольный целочисленный s-вектор с упорядоченными компонентами $1 = m_1^0 < m_2^0 < \ldots < m_r^0 \leqslant s$, называемый (истинным) шаблоном (связей); $Q^0 = (q^0_{j_1, \ldots, j_r, j_{r+1}})$, $j_1, \ldots, j_{r+1} \in \mathcal{A}$ — некоторая $(r+1)$-мерная стохастическая матрица.

Примем обозначения: $X_n = (x_1, \ldots, x_n) \in \mathcal{A}^n$ — наблюдаемый временной ряд длительностью $n > s$; $J_s \in \mathcal{A}^s$ — мультииндекс;

$$S_t(X_n;\, M_r) = (x_{t+m_1-1}, \ldots, x_{t+m_r-1}) \colon \mathcal{A}^n \times \mathcal{M} \to \mathcal{A}^r$$

— селектор r-го порядка с параметрами $M_r \in \mathcal{M}$, $t \in \{1, \ldots, n-s+1\}$, $\mathbf{I}(A) \in \{0, 1\}$ — индикатор события A;

$$\nu_{r+1}(J_{r+1};\, M_r) = \sum_{t=1}^{n-s} \mathbf{I}\{S_t(X_n;\, M_r) = J_{r+1}\}$$

— частота $(r+1)$-граммы J_{r+1} для шаблона $M_{r+1} = (M_r, s+1)$.

Теорема 1. *Если $M_r = M_r^0$ и $\min_{J_r \in \mathcal{A}^r} \nu_{r+1}(J_{r+1} \cdot\, ;\, M_r) > 0$, то условная оценка максимального правдоподобия (ОМП) для Q^0 имеет вид*: $\widehat{Q} = \widehat{Q}(M_r) = (\widehat{q}_{J_{r+1}}(M_r))$,

$$\widehat{q}_{J_{r+1}}(M_r) = \nu_{r+1}(J_{r+1};\, M_r) / \nu_{r+1}(J_{r+1} \cdot\, ;\, M_r), \quad J_{r+1} \in \mathcal{A}^{r+1},$$

где $\cdot$ означает суммирование по этому индексу по $\mathcal{A}$.

Теорема 2. *ОМП шаблона M_r^0 выражается через информационный критерий*: $\widehat{M}_r = \arg\max_{M_r \in \mathcal{M}} \widehat{I}_{r+1}(M_{r+1})$, *где для информационного функционала $\widehat{I}_{r+1}(M_{r+1})$ получено явное выражение и установлена асимптотическая нормальность при $n \to \infty$.*

Теорема 3. *Если* ЦМ(s, r) *стационарна и шаблон* $M_r^0 \in \mathcal{M}$ *обладает свойством индентифицируемости, то ОМП* $\widehat{M}_r$, $\widehat{Q}$ *— состоятельны: при* $n \to \infty$

$$\widehat{M}_r \xrightarrow{\mathsf{P}} M_r^0, \qquad \widehat{Q} \xrightarrow{\mathsf{P}} Q^0.$$

Вычислительная сложность алгоритма идентификации модели ЦМ(s, r) не превосходит $\mathcal{O}(N^{r+1}s^{r-1})$. Численные результаты подтверждают работоспособность алгоритма.

3. Использование модели Джекобса–Льюиса

Пусть дискретный временной ряд определятся моделью Джекобса–Льюиса [6]:

$$x_t = \mu_t x_{t-\eta_t} + (1-\mu_t)\xi_t, \quad t > s, \tag{2}$$

где $\{\xi_t : t > s\}$, $\{\eta_t : t > s\}$, $\{\mu_t : t > s\}$, $\{x_1, \dots, x_s\}$ — независимые в совокупности случайные величины со следующими вероятностными распределениями:

$$\begin{aligned}
&\mathsf{P}\{\xi_t = i\} = \pi_i, \quad i \in \mathcal{A}, \quad t > s, \quad \textstyle\sum_{i\in\mathcal{A}}\pi_i = 1;\\
&\mathsf{P}\{\eta_t = j\} = \lambda_j, \quad j \in \{1, \dots, s\}, \quad t > s, \quad \textstyle\sum_{j=1}^{s}\lambda_j = 1, \quad \lambda_s \neq 0;\\
&\mathsf{P}\{\mu_t = 1\} = 1 - \mathsf{P}\{\mu_t = 0\} = \rho, \quad t > s,\\
&\mathsf{P}\{x_1 = i\} = \dots = \mathsf{P}\{x_s = i\} = \pi_i, \quad i \in \mathcal{A}.
\end{aligned} \tag{3}$$

Отметим, что в [6] изучены лишь вероятностные свойства временного ряда x_t, статистические задачи не решались.

Теорема 4. *Временной ряд* x_t, *определяемый* (2), (3), *является однородной цепью Маркова порядка* s *с матрицей вероятностей переходов:* $P = (p_{i_1, \dots, i_s, i_{s+1}})$,

$$p_{i_1, \dots, i_s, i_{s+1}} = (1-\rho)\pi_{i_{s+1}} + \rho\sum_{j=1}^{s}\lambda_j\delta_{i_{s-j+1}, i_{s+1}}, \quad i_1, \dots, i_{s+1} \in \mathcal{A}, \tag{4}$$

где δ_{jk} *— символ Кронекера.*

Следствие 1. *Логарифмическая функция правдоподобия имеет вид:*

$$l(\pi, \lambda, \rho) = \sum_{t=1}^{s}\ln\pi_{x_t} + \sum_{t=s+1}^{n}\ln\Big((1-\rho)\pi_{x_t} + \rho\sum_{j=1}^{s}\lambda_j\delta_{x_{t-j}, x_t}\Big). \tag{5}$$

Примем обозначения: $P(\pi, \lambda, \rho)$ — матрица, элементы которой вычислены по (4); $\widehat{P}$ — эмпирическая матрица вероятностей переходов, вычисленная по наблюдаемому временному ряду X_n;

$$S(P) = \sum_{i_1, \dots, i_{s+1}\in\mathcal{A}} p^2_{i_1, \dots, i_s, i_{s+1}}$$

— сумма квадратов элементов матрицы P; $F(P)$ — сумма квадратов элементов матрицы P, для которых выполнено $\{i_1 = \dots = i_s = i_{s+1}\}$ или $\{i_1 \neq i_{s+1}, \dots, i_s \neq i_{s+1}\}$

Теорема 5. *Если* $\rho \neq 1$, *то при увеличении длины наблюдаемого временного ряда* $n \to \infty$ *состоятельными оценками для параметров модели* (2), (3) *являются статистики* $\widetilde{\pi}$, $\widetilde{\lambda}$, $\widetilde{\rho}$:

$$\widetilde{\pi}_i = \sum_{t=1}^{n}\delta_{x_t, i}/n, \quad i \in \mathcal{A}; \quad \widetilde{\rho} = \arg\min_{\rho\in[0,1)} F\Big(\widehat{P} - P(\widetilde{\pi}, \rho, \lambda)\Big);$$

$$\widetilde{\lambda} = \arg\min S\Big(\widehat{P} - P(\widetilde{\pi}, \widetilde{\rho}, \lambda)\Big),$$

причем получены явные формулы для вычисления $\widetilde{\lambda}$ *и* $\widetilde{\rho}$.

Эти оценки $(\widetilde{\pi}, \widetilde{\lambda}, \widetilde{\rho})$ используются в качестве начального приближения при итерационном вычислении ОМП $(\widehat{\pi}, \widehat{\lambda}, \widehat{\rho})$.

Согласно (2), (3), гипотеза $H_0 = \{\{x_t\}$ есть РРСП$\}$ допускает эквивалентное представление: $H_0 = \{\rho = 0,\ \pi_i = N^{-1},\ i \in \mathcal{A}\}$. Построен тест проверки гипотез H_0, $H_1 = \overline{H}_0$, основанный на статистике обобщенного отношения правдоподобия $\lambda_n = 2\big(l(\widehat{\pi}, \widehat{\lambda}, \widehat{\rho}) + n\ln N\big)$ и имеющий асимптотический (при $n \to \infty$) размер $\varepsilon \in (0, 1)$:

$$d = d(x_1, \dots, x_n) = \{0,\ \lambda_n < \Delta_\varepsilon;\ \ 1,\ \lambda_n \geqslant \Delta_\varepsilon\}, \tag{6}$$

где Δ_ε — квантиль уровня ε хи-квадрат распределения с N степенями свободы.

Численные результаты подтверждают достаточную точность идентификации и принятия решений.

Список литературы

[1] *Алферов А. П., Зубов А. Ю. Кузьмин А. С., Черемушкин А. В.* Основы криптографии. М.: Гелиос, 2001.

[2] *Харин Ю. С., Берник, Матвеев Г. В., Агиевич С. В.* Математические и компьютерные основы криптологии. Минск: Новое знание, 2003.

[3] *Raftery A. E.* A model for high-order Markov chains. Journal of the Royal Statistical Society, B, vol. 47, no. 3, 1985, pp. 528–539.

[4] *Тихомиров М. Н., Чистяков В. П.* О двух статистиках типа хи-квадрат, построенных по частотам цепочек состояний сложной цепи Маркова. Дискретная математика, т. 15, вып. 3, 2003, с. 149–159.

[5] *Харин Ю. С.* Цепи Маркова с r-частичными связями и их статистическое оценивание. Доклады НАН Беларуси, т. 48, № 1, 2004, с. 40–41.

[6] *Jacobs P. A., Lewis P. A. W.* Discrete time series generated by mixtures, I, II. Journal of the Royal Statistical Society, no. 1, 2, 1978, pp. 95–105, 222–228.

О величине простых делителей чисел вида $p-1$

М. А. Черепнёв

Вопрос о том, являются или нет задачи дискретного логарифмирования и Диффи — Хеллмана полиномиально эквивалентными, пока открыт. В статье [8] этот вопрос сведен к оценке следующей теоретико-числовой функции.

Пусть

$$m=\prod_{i=1}^{r} {p_i}^{\alpha_i}.$$

Рассмотрим разложение на простые множители чисел p_i-1, $i=1,\ldots,r$. Из каждого из этих простых вычтем единицу и снова разложим на простые, и так далее. Получившееся разветвление назовем деревом числа m, а встречающиеся в нем простые числа его узлами. Обозначим $s=s(m)$ длину наибольшей ветви дерева числа m.

Для оценки $s(m)$ для почти всех m можно применить результаты о количестве и величине различных простых делителей чисел вида $p-1$. Из оценок снизу для этих величин можно надеяться в дальнейшем получить оценку для $s(m)$ [9].

В своих работах Харди и Рамануджан [3], Туран [6] показали, что за исключением $\overline{o}(x)$ натуральных значений n, $n\leqslant x$, выполняются неравенства

$$(1-\varepsilon)\ \ln\ln n<v(n)<(1+\varepsilon)\ \ln\ln n,$$

где $v(n)$ — количество различных простых делителей числа n. Аналогичный результат для чисел вида $p-1$, где p — простое число, получил Эрдёш [1]. В данной заметке похожий результат получен для количества простых делителей чисел $p-1$, больших некоторой растущей границы. Заметим, что ввиду очевидной оценки $s(n)\leqslant\log n$ маленькие простые делители не оказывают влиянияна величину $s(m)$.

Обозначим $v_{>t}(n)$ количество различных простых делителей числа n, больших t. Обозначим $N(M)$ — количество элементов в множестве M.

Теорема. *Для любого* $\varepsilon\in(0,\ 1/3)$ *и* $t=\exp\bigl(\ln\ x^{(1-\varepsilon)/(2e)}\bigr)$

$$N\Big(p\leqslant x\;\Big|\;v(p-1)\in[(1-\varepsilon)\ln\ln x,\,(1+\varepsilon)\ln\ln x],\quad v_{>t}(p-1)>\frac{1}{2}v(p-1)\Big)=$$
$$=\frac{x}{\ln x}+\overline{o}\Big(\frac{x}{\ln x}\Big)\,.$$

Доказательство. Рассмотрим множество

$$M=\{p\leqslant x\mid v(p-1)\in[(1-\varepsilon)\ln\ln x,\;\;(1+\varepsilon)\ln\ln x]\}.$$

По теореме 7.2 на странице 188 [7] $N(M)=x/\ln x+\overline{o}(x/\ln x)$. Поэтому для доказательства нашей теоремы достаточно установить, что

$$N\Big(p\in M\;\Big|\;v_{\leqslant t}(p-1)>\frac{1}{2}v(p-1)\Big)=\overline{o}\Big(\frac{x}{\ln x}\Big).$$

Как доказано на странице 189 [7], число простых p не превосходящих x, для которых все простые делители $p-1$ не превосходят $y=e^{\ln x/\ln\ln x}$, есть $\overline{o}(x/\ln x)$. Значит, из нашего рассмотрения их можно отбросить. Поэтому рассматриваемое количество можно оценить так:

$$\sum N\Big(p\leqslant x\;\Big|\;p,\;\frac{p-1}{n}\;\text{— простые}\Big)<$$
$$<\sum N\Big(r<\frac{x}{n}\;\Big|\;r,\;rn+1\;\text{— простые}\Big)\,,$$

где суммирование ведется по всем натуральным n, удовлетворяющим неравенствам $n<x/y$, $v_{\leqslant t}(p-1)>v(p-1)/2$, $v(p-1)\in[(1-\varepsilon)\ln\ln x,$ $(1+\varepsilon)\ln\ln x]$. Слагаемые в последней сумме можно оценить по теореме 4.2 на странице 54 [7] при $s=2$. Получим

$$<\sum\frac{xn}{n\ln^2\frac{x}{n}\varphi(n)}<cx\sum\frac{1}{\varphi(n)\ln^2\frac{x}{n}}<\frac{cx}{\ln^2 y}\sum\frac{1}{\varphi(n)}.$$

Здесь суммирование ведется по тем же натуральным n. Далее воспользуемся мультипликативностью функции Эйлера и известными оценками сумм по простым числам (см. [7], теорема 4.1 на странице 28).

$$\leqslant\frac{cx}{\ln^2 y}\sum\frac{1}{\Big(k-\frac{1-\varepsilon}{2}\ln\ln x\Big)!\,\Big(\frac{1-\varepsilon}{2}\ln\ln x\Big)!}\times$$
$$\times\Big(\sum_{p<x}\sum_{i=1}^{\infty}\frac{1}{\varphi(p^i)}\Big)^{k-(1-\varepsilon)/2\ln\ln x}\Big(\sum_{p\leqslant t}\sum_{i=1}^{\infty}\frac{1}{\varphi(p^i)}\Big)^{(1-\varepsilon)/2\ln\ln x}<$$

Здесь суммирование ведется по $k\in[(1-\varepsilon)\ln\ln x,\,(1+\varepsilon)\ln\ln x]$.

$$<\frac{cx}{\Big(\frac{1-\varepsilon}{2}\ln\ln x\Big)!\ln^2 y}\times$$
$$\times\sum_{\substack{k\in[(1-\varepsilon)/2\ln\ln x,\\(1+3\varepsilon)/2\ln\ln x]}}\frac{1}{k!}(\ln\ln x+c_1)^k(\ln\ln t+c_2)^{(1-\varepsilon)/2\ln\ln x}\leqslant$$

для некоторых абсолютных постоянных c_1, c_2. Применяя формулу Стирлинга, получим для $\varepsilon_1=(1-3\varepsilon)/2>0$ следующую оценку:

$$\leqslant\frac{cx}{\ln^2 y}\Big(\frac{2e}{1-\varepsilon}\frac{\ln\ln t+c_2}{\ln\ln x}\Big)^{(1-\varepsilon)/2\ln\ln x}\sum_{k\leqslant(1-\varepsilon_1)\ln\ln x}\frac{(\ln\ln x+c_1)^k}{k!}$$

Теперь воспользуемся оценкой для получившейся суммы со страницы 191 [7]. Получим

$$\leqslant\frac{cx\ln\ln^2 x\ln^{1-\delta}x}{\ln^2 x}\Big(\frac{2e}{1-\varepsilon}\frac{\ln\ln t+c_2}{\ln\ln x}\Big)^{1/2\ln\ln x}$$

Последнее при выбранном в условии t есть $\overline{o}(x/\ln x)$. □

В заключение докажем одну простую лемму, которая может оказаться полезной в будущем для оценок, подобных тем, что мы только что рассмотрели.

Лемма. *Для некоторой абсолютной константы c*

$$N\Big(n\leqslant x\;\Big|\;\varphi(n)>\frac{cn}{\ln\ln\ln n}\Big)=x+\overline{o}(x)$$

Доказательство. Как уже отмечалось в начале,

$$N(n\leqslant x|\;v(n)\in[(1-\varepsilon)\ln\ln x,\,(1+\varepsilon)\ln\ln x])=x+\overline{o}(x).$$

Докажем, что для этих n выполняется нужная оценка.

$$\varphi(n)=n\prod_{p|n}\Big(1-\frac{1}{p}\Big)\geqslant n\prod_{i=1}^{v(n)}\Big(1-\frac{1}{p_i}\Big)=n\prod_{p\leqslant p_{v(n)}}\Big(1-\frac{1}{p}\Big)>$$

По теореме 4.1 на странице 28 [7] для некоторой абсолютной константы c_1 получим оценку

$$> c_1 \frac{n}{\ln\, p_{v(n)}} > \frac{c_2 n}{\ln(v(n)\, \ln\, v(n))} > \frac{cn}{\ln\, \ln\, \ln\, n}. \quad \square$$

Список литературы

[1] *Erdős P.* On the normal number of prime factors of $p-1$ and some related problems concerning Euler's φ-function. Quart. J. Oxford, 6(1935), 205–213.

[2] *Boer B.* Diffie-Hellman is as strong as discrete log for certain primes. Lect. Notes. Comp. Sci., 403(1988), 530–540.

[3] *Hardy G. H., Ramanujan S.* The normal number of prime factors of a number n. Quart. J. of Math., 48(1917), 76–92.

[4] *Maurer U. M.* Towards the equivalence of breaking the Diffie-Hellman protocol and computing discrete logarithms. Crypto'94, 271–281.

[5] *Sacurai K., Shizuya H.* Relationships among the computational powers of breaking discrete log cryptosystems. Eurocrypt'95, 341–355.

[6] *Turán P.* On a theorem of Hardy and Ramanujan. J. Lond. Math. Soc., (2), 30(1929), 93–111.

[7] *Прахар К.* Распределение простых чисел. — М.: Мир, 1967, 511 с.

[8] *Черепнёв М. А.* О связи сложностей задач дискретного логарифмирования и Диффи — Хеллмана. М., Дискр. мат., с. 22–30.

[9] *Черепнёв М. А.* Некоторые свойства больших простых делителей чисел вида $p-1$. МаБИТ-03, с. 218–220.

Секция «Информационная безопасность компьютерных систем»

Враждебные многоагентные системы

А. А. Грушо, Е. Е. Тимонина

В последние годы наблюдается значительный прогресс в области создания распределенных систем. Как правило, интеграция распределенных систем осуществляется с помощью Middleware (MW). В основе MW лежит многоагентная система, осуществляющая взаимодействие с приложениями и позволяющая распределенной системе решать единые задачи. Согласно спецификации FIPA программный агент — это некоторая вычислительная сущность, способная к автономному поведению, то есть программный агент может делать выбор из некоторого набора действий, когда сталкивается с задачей принятия решений, относящейся к его области действий. Помимо этого такая сущность рассматривается как часть сообщества подобных сущностей, которые спроектированы, чтобы взаимодействовать друг с другом для достижения общих целей. Агенты должны иметь возможность взаимодействовать друг с другом.

В работе рассматривается возможность создания враждебной многоагентной архитектуры в распределенной компьютерной системе. В работе приведены результаты исследований следующих задач.

1. Защита агента от средств защиты.
2. Взаимодействие агентов между собой внутри отдельного компьютера.
3. Защита одних агентов с помощью других от средств защиты.
4. Взаимодействие враждебной многоагентной системы с внешней средой.
5. Разведка компьютерной среды.
6. Развитие враждебной многоагентной системы.
7. Реставрация враждебной многоагентной системы при сбоях и разрушениях.
8. Нарушение целостности информационных ресурсов.
9. Блокирование информационных ресурсов и программ.

В основы модели враждебной многоагентной системы положены две разработанные и реализованные на практике модели. Первая —

Работа поддержана РФФИ, проект 04-01-00089.

это модель невлияния, которая разработана американскими учеными и частично авторами этой работы. Данная модель позволяет сделать вывод о том, что при определенных условиях возможно существование многоагентной распределенной враждебной системы, которая «невидима» для средств защиты компьютерной системы. Вторая модель, положенная в основу данной работы, называется Open Agent Architecture (OAA), разработана несколько лет назад в американском институте SRI [9].

Пусть компьютерная среда разбита на два уровня High и Low. Если субъект на уровне High может выполнять любые действия и «видеть» все действия на уровне Low, но любой субъект на уровне Low не может наблюдать какие-либо следы деятельности на уровне High, то система удовлетворяет условиям невлияния.

Если враждебный агент находится на уровне High, а средства защиты на уровне Low и выполняются условия невлияния, то агент «невиден» для средств защиты [1].

Условия невлияния могут быть реализованы в компьютерной среде, например, с помощью:

1) «закладки» в процессоре;

2) «закладки» в ядре операционной системы.

Скрытый канал — это способ передачи информации, который не выявляется средствами защиты.

Агенты в компьютерной системе могут связываться между собой с помощью скрытых каналов. Примеры таких скрытых каналов приведены в работах [5], [8]. Особо отметим, что каналы в шине могут наблюдаться только с помощью процессоров, которых в РФ нет.

Если множество враждебных агентов на уровне High связано скрытыми каналами, а средства защиты находятся на уровне Low и выполняются условия невлияния, то распределенная сеть враждебных агентов «невидима» для средств защиты [2].

Для решения задач, перечисленных выше, сеть «невидимых» агентов должна обладать интеллектуальными функциями.

В качестве прообраза враждебной многоагентной архитектуры (ВМА) взята Open Agent Architecture, разработанная SRI. Данная архитектура позволяет придать ВМА интеллектуальные функции, позволяющие объяснить, как решаются перечисленные выше задачи.

Пусть сеть враждебных агентов «невидима» для средств защиты. Выделяются три типа агентов:

1) агенты-посредники;

2) мета-агенты;

3) агенты интерфейса с внешними сетями.

Первый тип агента — агент посредник. Агент посредник представляет собой специализированного служебного агента, который отвечает за координацию взаимодействия агентов, их связь и кооперацию. В некоторых случаях посредник обеспечивает хранение данных для других типов агентов, то есть предоставляет им функции «black board» для их взаимодействия. Посредники должны быть связаны между собой в некоторый кластер. В частном случае это может быть иерархическая структура.

Следующим типом агентов является мета-агент. Этот тип агента ориентирован на конкретный набор функций в ВМА. Функции этого агента могут зависеть от специфического домена, но могут также и не зависеть от каких либо доменов. Мета-агенты исполняют роль ассистентов посредников при координации деятельности всех агентов. Если агент посредник обладает набором независимых от домена координационных функций, то мета-агенты содержат предметно-ориентированную информацию, связанную либо с заданным доменом, либо с заданным видом приложений, либо с заданным видом деятельности.

Особо необходимо выделить агентов, реализующих функции интерфейсов. Эти агенты не являются посредниками, они реализуют интерфейсы с глобальными сетями, между узлами локальных сетей и т. д. Особенностью этих агентов является то, что при реализации модели невлияния данные агенты реализуют функции скрытых каналов, использующие открытые коммуникации нижнего уровня. В терминологии многоуровневой политики эти агенты реализуют доверенные процессы незаметного спуска информации на нижний уровень для последующей скрытой передачи по общедоступным каналам.

Всех агентов, которые не являются посредниками, будем называть агентами-клиентами. Каждый клиент соединен, по крайней мере, с одним посредником, который мы будем называть посредником-родителем для данного клиента. С помощью связи агент-клиент информирует посредника-родителя о сервисах, которые он может реализовывать, о среде, которую он видит, о помощи, которую ему необходимо оказывать. По аналогии с OAA для связи клиентов с посредниками необходим язык межагентной связи. Посредник получает запросы от клиента, отвечает ему или контролирует статус клиента. Для реализации запроса посредник задействует связь с другими посредниками и мета-агентами. Для создания агентов используются специальные библиотеки агентов. С помощью этих библиотек создаются, настраиваются новые агенты, осуществляется уничтожение агентов. Коды агентов могут поступать извне через агентов интерфейса и коммуникационную сеть посредников.

Основные идеи ВМА состоят в следующем.

1. Новый агент устанавливает связь с посредником, осуществляет поиск в окружающей среде информации по заданным признакам, создает образ среды и передает посреднику информацию об окружающей среде и о тех действиях, которые он может реализовать в этой среде. Данная информация через посредника передается мета-агентам, которые из фрагментов составляют некоторую общую картину компьютерной среды, определяют возможные действия в этой среде, определяют структуру защиты и организуют через посредников и прикладных агентов нейтрализацию функций защиты, в частности, «невидимость» ВМА для системы защиты. Особую роль играют агенты, обнаруживающие в окружающей среде каналы во внешнюю среду. Мета-агенты обеспечивают таких агентов информацией, позволяющую установить связь с другими агентами через найденные каналы во внешнюю среду. При установлении связи с внешними приложениями агент становится агентом интерфейса.

2. Интерпретация и выполнение заданий для ВМА является распределенным процессом.

3. Запрос одного прикладного агента может породить кооперацию и взаимодействие среди многих агентов ВМА.

Кооперация среди агентов ВМА достигается через передачу сообщений на общем языке, доступном пониманию агентами. Обычная процедура кооперации состоит из двух шагов.

1. Агенты через посредника снабжают мета-агента информацией о доступных сервисах.
2. Запросы на сервисы поступают от мета-агентов через посредников.

Посредники координируют действия агента при достижении заданной цели (например, взять информацию, передать ее через посредников агенту, имеющему связь с внешним нарушителем безопасности).

Структура языка взаимодействия агентов основана на понятии события. Активность всех агентов, а также и коммуникаций между агентами строится вокруг передачи и обработки событий. При коммуникациях события составляют содержание сообщений между агентами. События являются целями для действий агентов. Каждое событие имеет тип, множество параметров и содержание. Допустимые содержания и значения параметров могут различаться в зависимости от типа события.

Рассмотрим потенциально возможную схему создания ВМА в компьютерной или распределенной компьютерной системе.

1. Должен существовать сигнал активизации агента посредника. Этот сигнал может представлять собой «логическую бомбу» или сигнал извне, полученный по каналам связи из внешней среды. Посредник может быть заложен производителем в процессоре компьютерной системы. Однако возможны другие возможности скрытого от защиты местоположения посредника.

2. Активизированный посредник по шинам или другим каналам в системе передает запросы или сигналы активизации другим посредникам, которые могут присутствовать в системе. При получении ответа создается распределенная сеть посредников. В случае выполнения условия невлияния для каждого посредника и скрытых каналов взаимодействия между ними полученная распределенная сеть посредников будет удовлетворять условиям невлияния, то есть будет «невидима» для средств защиты.

3. Одновременно посредник (сеть посредников) должен создать хотя бы одного мета-агента.

4. Сеть посредников ищет канал во внешнюю среду и при наличии такого канала использует один из языков, заложенных в данные мета-агента, для связи с внешней средой.

5. Для исследования окружающей среды с помощью мета-агентов и посредников создается сеть агентов.

Исследования SRI показывают, что враждебная многоагентная система, описанная выше, может быть реализована и функционировать с тем уровнем интеллекта, который описан в работе [9].

Первая задача, из перечисленных в начале статьи, может быть решена с помощью «закладки» в процессоре, в котором выполнены условия невлияния, или с помощью агента, внесенного в ядро операционной системы, так что он не допускает выполнение любых системных вызовов, которые могут быть использованы для контроля среды. При этом агент перехватывает эти вызовы и модифицирует ответы на них.

Защита каналов внутри компьютера, связывающих агентов между собой, была рассмотрена ранее.

Для контроля над средой можно, например, реализовать режим ортогонального времени. То есть в то время, когда запускаются программы и коды агентов, все остальные задачи и процессы приостанавливают свою работу. В условиях, когда посредник не дает средствам защиты обращаться к разделам памяти, где находятся агенты, создание ортогонального времени эквивалентно условиям невлияния. Поэтому для решения задачи контроля среды и нейтрализации защиты активированный посредник и мета-агент ищут признаки планировщика и программы распределения памяти. Контроль запуска этих элементов операционной системной эквивалентен созданию ортогонального времени. Другим уязвимым местом ВМА является возможность функционирования про-

грамм защиты сканирующих среду и работу процессоров. Поэтому следующим шагом установления контроля над средой является выявление по признакам, полученных от мета-агентов подобных сканеров, и нейтрализация их работы путем имитации необходимых ответов, передающихся этим сканерам. Заключительной фазой установления контроля за средой является формирование доменов, защищенных от средств защиты системы. Эффективность решения этой задачи определяется интеллектом мета-агентов и контролем посредника над запуском программ. В том случае, если посредник является закладкой в процессоре, а мета-агент обладает достаточно большими интеллектуальными возможностями, задачи защиты от средств защиты и контроля среды могут быть эффективно решены.

Как было отмечено выше, агент интерфейса должен реализовать взаимодействие с глобальными сетями и агентами в них. Чаще всего на границе с глобальными сетями устанавливаются сильные системы защиты (например, IPsec и др.). Можно показать, что несмотря на любые применяемые системы защиты, включая шифрование, можно построить скрытые каналы для взаимодействия агентов интерфейса с агентами в глобальной сети [3], [4], [6], [7].

Разведка среды осуществляется, в первую очередь, с помощью взаимодействия мета-агентов через посредников. Мета-агент с помощью посредника создает сканирующего агента (также мета-агента), защищенного от средств защиты. Сканирующий агент выделяет (распознает) элементы среды, в которую он помещен. Событием является распознавание элемента среды или неуспех задачи распознавания. Данные о событиях передаются исходным мета-агентам, которые создают условный портрет среды.

Другим вариантом сканирования среды является создание сканирующих агентов (предметно-ориентированных) из внешней среды.

В распределенной сети только часть узлов может быть связана в ВМА. Одна из функций, которую может выполнять ВМА, это захват других узлов и компьютеров в сети или создание новых агентов, то есть ВМА может развиваться. Развитие ВМА связано с возможностью активизации хотя бы одного посредника в узле, не захваченном ВМА.

Узлы, активизированные из внешней среды, могут строить мета-агента с помощью передачи соответствующих кодов из ВМА по скрытым каналам. Кроме того, мета-агенту может быть передана вспомогательная информация о портрете защиты в захватываемом компьютере.

Протоколы передачи данных по скрытому каналу при взаимодействии агентов между собой не предполагают гарантированную доставку. Это связано с возможностью отключения отдельных узлов, сбоями

в программно-аппаратной среде и другими причинами. Часть агентов, которые выполнили свою функцию, например, не являющиеся предметно-ориентированными и которые заменены на предметно-ориентированные, должны быть уничтожены. Кроме того, в целях безопасности ВМА должны уничтожаться все следы деятельности агентов при использовании ортогонального времени. Контроль за жизнью и смертью агентов осуществляет посредник соответствующих агентов, а команды на уничтожение даются мета-агентом. Для интеллектуальных сканирующих агентов иногда выгодно уничтожать их, чем прятать в ограниченные объемы памяти, защищенные посредником. Процесс восстановления разрушенной части ВМА эквивалентен процессу созданию ВМА с той лишь разницей, что в мета-агенте остаются схемы или другие следы, позволяющие ускорить такое восстановление «невидимо» для средств защиты. Следует отметить, что эффективным средством защиты от механизмов защиты системы является «сон» агентов в защищенных доменах в период работы средств защиты.

Атаки на целостность информационных ресурсов и программ могут быть реализованы благодаря контролю ВМА над средой. При этом возможно три сценария:

1) модификация или уничтожение информационных ресурсов непосредственно в компьютерной среде;
2) модификация или уничтожение данных при резервировании;
3) хранение признаков уничтоженных или модифицированных объектов с целью их модификации или уничтожения при восстановлении их из резервных копий.

Возможность блокирования информации связана с возможностью создания временно «невидимых» файлов и программ. В качестве вывода отметим, что исследования SRI и другие исследования полностью подтверждают возможность реализации рассмотренной модели.

Список литературы

[1] *Грушо А. А.*, *Тимонина Е. Е.* Двойственность многоуровневой политики безопасности // Тез. докл. конф. «Методы и технические средства обеспечения безопасности информации». СПб: СПбГТУ, 2000. С. 40–41.

[2] *Грушо А. А.*, *Тимонина Е. Е.* Модель невлияния для сети // Обозрение прикладной и промышленной математики. М.: ТВП, 2000. Т. 7. Вып. 1 С. 185–187.

[3] *Грушо А. А.*, *Тимонина Е. Е.* Языки в скрытых каналах // Материалы XXX юбилейной международной конференции и I международной конференции молодых ученых «Информационные технологии в науке, об-

разовании, телекоммуникации, бизнесе. IT+SE'2003 (майская сессия)» (19–28 мая). Ялта-Гурзуф, 2003. С. 181–184.

[4] *Грушо А. А.*, *Тимонина Е. Е.* Оценка времени, требуемого для организации скрытого канала // Дискретная математика. 2003. Т. 15 Вып. 2. С. 40–46.

[5] *Тимонина Е. Е.* Скрытые каналы (обзор) // Jet Info: изд-во компании «Джет Инфо Паблишен», 2002. 14(114). С. 3–11.

[6] *Тимонина Е. Е.* Скрытые каналы, построенные с помощью модуляции протоков адресов пакетов // Успехи современного естествознания. М.: «Академия естествознания», 2004. № 5.

[7] *Grusho A.*, *Timonina E.* Construction of the Covert Channels // International Workshop «Information Assurance in Computer Networks. Methods, Models, and Architectures for Network Security» (MMM-ACNS-2003). St. Petersburg: Springer, 2003. LNCS 2776. P. 428–431.

[8] A Guide to Understanding Covert Channel Analysis of Trusted Systems, National Computer Security Center. NCSC-TG-030. Ver. 1, 1993.

[9] *Martin D. L.*, *Cheyer A. J.*, *Moran D. B.* The Open Agent Architecture: A Framework for Building Distributed Software Systems.

Многоагентные модели противоборства злоумышленников и системы защиты в сети Интернет

И. В. Котенко

Введение

Вопросы моделирования обеспечения информационной безопасности активно исследуются на протяжении более чем тридцати лет. Разработано большое количество формальных и неформальных моделей отдельных механизмов защиты. Однако существует очень мало работ, формализующих комплексный антагонистический характер обеспечения информационной безопасности как сложного организационно-технического процесса, в том числе рассматривающих задачу обеспечения информационной безопасности как комплексную задачу организационного и технического компьютерного противоборства между системами защиты информации и системами компьютерного нападения злоумышленников, а также базирующихся на исследовательском моделировании указанного комплекса процессов.

Это объясняется сложностью данной предметной области. Хотя исследователи в состоянии представить отдельные механизмы защиты, понимание системы защиты как единой холической системы, зависящее от учета множества взаимодействий между отдельными процессами кибер-противоборства и динамического характера этих процессов и компонент информационных систем, чрезвычайно затруднено.

Особенно это справедливо с учетом наблюдаемой в настоящее время эволюции Интернет в свободную децентрализованную распределенную среду взаимодействия огромного числа кооперирующихся и антагонистических программных агентов, обменивающихся между собой и с пользователями-людьми большими объемами информации и услуг (сервисов).

Работа выполняется при поддержке РФФИ (проект N 04-01-00167) и программы фундаментальных исследований ОИТВС РАН (контракт N 3.2/03).

В работе рассматривается агентно-ориентированный подход к формальному представлению и моделированию противоборства злоумышленников и систем защиты в виде антагонистического взаимодействия команд программных агентов на примере моделирования реализации DDoS-атак и защиты от них.

1. Предлагаемый подход к моделированию

Использование основанного на многоагентных технологиях моделирования процессов обеспечения информационной безопасности в сети Интернет предполагает, что кибернетическое противоборство представляется в виде взаимодействия различных команд программных агентов. Агрегированное поведение системы проявляется посредством локальных взаимодействий отдельных агентов в динамической среде, задаваемой посредством модели сети Интернет.

Выделяется, по крайней мере, две команды агентов, воздействующих на компьютерную сеть, а также друг на друга:

1) команда агентов-злоумышленников;

2) команда агентов защиты.

Задача многоагентного моделирования процессов кибернетического противоборства представляется как моделирование коэволюционного антагонистического взаимодействия команды агентов-злоумышленников и агентов защиты.

Цель команды агентов-злоумышленников состоит в определении уязвимостей компьютерной сети и системы защиты и реализации заданного перечня угроз информационной безопасности (конфиденциальности, целостности и доступности) посредством выполнения распределенных скоординированных атак.

Цель команды агентов защиты состоит в защите сети и собственных компонентов от атак.

Команда агентов-злоумышленников реализует развитые стратегии, включающие сбор информации о системе — цели нападения, обнаружение уязвимостей и используемых средств защиты, моделирование способов преодоления защиты, подавление, обход или обман средств защиты (например, посредством реализации «растянутого» во времени скрытого сканирования, выполнения отдельных скоординированных действий (атак) из нескольких различных источников, вместе составляющих сложную многофазную атаку и др.), использование уязвимостей и получение доступа к ресурсам, повышение полномочий, реализацию определенной угрозы, скрытие следов своей деятельности и создание «черных ходов» для использования их для последующего вторжения.

Примером автоматической стратегии является поражение сети Интернет, возникающие в результате распространения сетевых вирусов и червей, в том числе недавние эпидемии, высвечивающиеся тенденцию сращивания вирусных и спам-технологий и формирования объединенной, мотивированной сети агентов-злоумышленников.

Команда агентов защиты выполняет в реальном времени последовательность следующих действий:

- реализация механизмов защиты, соответствующих установленной политике безопасности (в том числе проактивного препятствования вторжениям, блокирования атак, а также их обнаружения);
- сбор информации о состоянии защищаемой системы и анализ обстановки;
- предсказание намерений и возможных действий злоумышленников;
- заманивание злоумышленников с использованием ложных информационных компонентов с целью введения в заблуждение и уточнения их целей;
- непосредственное реагирование на вторжения, в том числе усиление критичных механизмов защиты;
- устранение последствий вторжения, выявленных уязвимостей и адаптация системы обеспечения информационной безопасности к последующим вторжениям.

Агенты различных команд соперничают для достижения противоположных намерений. Агенты одной команды сотрудничают для осуществления общего намерения (по реализации угрозы или по защите компьютерной сети). Предполагается, что соперничающие агенты осуществляют сбор информации из различных источников, оперируют нечеткими (вероятностными) знаниями, прогнозируют намерения и действия оппонента, оценивают возможные риски, пытаются обмануть друг друга, реагируют на действия оппонента и т. п. Предлагаемый подход к организации командной работы агентов базируется на совместном использовании элементов теории общих намерений [1], теории разделяемых планов [2] и комбинированных подходов [3].

Структура команды агентов описывается в терминах иерархии групповых и индивидуальных ролей. Листья иерархии отвечают ролям индивидуальных агентов, промежуточные узлы — групповым ролям. Механизмы взаимодействия и координации агентов базируются на трех группах процедур:

1) обеспечение согласованности действий;

2) мониторинг и восстановление функциональности агентов;

3) обеспечение селективности коммуникаций (для выбора наиболее «полезных» коммуникационных актов).

Спецификация иерархии планов действий осуществляется для каждой из ролей. Для каждого плана описываются: начальные условия, когда план предлагается для исполнения; условия, при которых план прекращает исполняться; действия, выполняемые на уровне команды, как часть общего плана. Для групповых планов явно выражается совместная деятельность.

Спецификация иерархии планов действий осуществляется в виде иерархии атрибутных стохастических формальных грамматик, связанных операцией подстановки [4]. Назначение ролей и распределение планов между агентами выполняется в два этапа: сначала план распределяется в терминах ролей, а потом каждой из ролей ставится в соответствие агент.

Команда агентов-злоумышленников эволюционирует посредством генерации новых экземпляров и типов атак с целью преодоления подсистемы защиты. Команда агентов защиты адаптируется к действиям злоумышленников путем формирования новых экземпляров механизмов и профилей защиты. Взаимодействие между агентами разных команд представляется как игра двух соперников, в которой целью агентов является поиск стратегии, которая максимизирует ожидаемый интегральный выигрыш в игре.

Для того, чтобы справится с гетерогенностью и распределенностью источников информации и используемых агентов предлагается применять основанный на онтологии подход и специальные протоколы для спецификации распределенного согласованного тезауруса понятий. Онтология предметной области обеспечения безопасности компьютерных сетей реализуется на базе стандартных языковых средств RDF или DAML+OIL. Проектирование и реализация рассмотренной многоагентной системы осуществляется на базе разработанного авторами проекта программного инструментария MASDK («Multi-agent System Development Kit» [5]) и (или) использования других инструментариев, например, JADE.

2. Онтологии атак и механизмов защиты

В разработанных онтологиях DDoS-атак и механизмов защиты понятия вышележащего уровня связываются с соответствующими понятиями смежного нижележащего уровня при помощи отношений трех видов: «Part of»; «Kind of»; «Seq of». Каждый пакет связан отношением «Example of» с конкретной реализацией.

Верхний уровень *онтологии DDoS-атак* составляют узлы, задающие множество программных средств реализации атак. На нижних уровнях специфицируются различные классы DDoS-атак.

Верхний уровень *онтологии механизмов защиты от DDoS-атак* составляют узлы, задающие уровни механизмов защиты (системный, сетевой, глобальный). На нижних уровнях онтологии эти узлы раскрываются на конкретные механизмы защиты.

Выделены три типа узлов, соответствующих *механизмам системного уровня*. *Инструменты сканирования* реализуют проверку наличия DDoS-агентов в файловой системе, а также сканирование портов, наиболее часто используемых атакующими. *Механизмы создания «узкого места»* направлены на реализацию на компьютерах «зомби», используемых для DDoS-атак, системного процесса, который снижает пропускную способность канала атаки. *Механизмы защиты посредством перемещения цели атаки* заключаются в изменении IP-адреса атакуемого хоста.

Механизмы сетевого уровня представлены следующими узлами:

1) механизмы, реализуемые на граничных маршрутизаторах (с подчиненными узлами — входящая фильтрация, исходящая фильтрация, MULTOPS);
2) механизмы, реализуемые на межсетевых экранах (предназначены для отсеивания пакетов, реализующих DoS-атаки);
3) активный мониторинг (для непрерывного наблюдения за состояниями сети, контроля TCP/IP-трафика и реагирования на критические ситуации);
4) уравновешивание нагрузки (на основе использования множества конфигурируемых входных и выходных очередей Web-сервера).

3. Команды агентов-злоумышленников и агентов защиты

Команда агентов реализации DDoS-атак состоит из «клиента», контролирующего команду «мастеров», которые, в свою очередь, контролируют множество «демонов» и передают им команды, присланные «клиентом» [6].

«Демоны» исполняют фактические действия по нападению на хост-жертву. В соответствии с процедурами обеспечения согласованности действий до начала реализации DDoS-атаки происходит формирование необходимого количества агентов, до их сведения доводятся их роли. Далее агенты сообщают о своей готовности и начинают активные действия в соответствии с заданной ролью. При достижении поставлен-

ной цели, обнаружении невозможности выполнить цель или выявлении нерелевантности цели, агент обязан сообщить этот факт оставшимся членам команды. При этих условиях выполняемый сценарий завершается, и должен быть активизирован другой сценарий.

Верхний уровень иерархии планов действий включает три этапа:

1) подготовительный,
2) основной,
3) заключительный.

Базовыми операциями подготовительного этапа являются разведка и установка агентов для последующей атаки. Содержание основного этапа состоит в реализации DDoS-атаки путем совместного действия агентов. Получив в результате цепочки сообщений адрес «жертвы», агенты-атакующие начинают процедуру поражения выбранного хоста. В это время агенты-разведчики ведут мониторинг состояния жертвы. При обнаружении ими успешности атаки, они доводят этот результат до всех агентов. В случае нерелевантности хоста (хост приобрел статус отключенного от сети) или невозможности его поражения, происходит либо окончание операции, либо, выбор новой цели для атаки. На заключительном этапе осуществляется сокрытие следов и установление «скрытых ходов».

Агенты защиты располагаются в наиболее на хостах, располагаемых в критических точках защищаемой сети. Команда агентов на хосте может состоять из одного или нескольких экземпляров агентов различных классов. *Агент-сенсор* (*АС*) осуществляет предварительную обработку поступающих на хост сообщений, фиксируя значимые для защиты информации события. *Агент идентификации и аутентификации* (*АИА*) ответствен за идентификацию источников сообщений и подтверждение их подлинности. *Агент разграничения доступа* (*АРД*) регламентирует доступ пользователей к ресурсам сети в соответствии с их правами. Агенты *АИА* и *АРД* обнаруживают несанкционированные действия по доступу к информационным ресурсам хоста и прерывают соединения и процессы обработки событий, отнесенные к числу несанкционированных. *Агент обнаружения вторжений* (*АОВ*) отвечает за обнаружение отдельных «подозрительных» событий или очевидных атак и принятие решений относительно реакции на данные события. *Интеллектуальные агенты обнаружения вторжений* (*ИАОВ*) реализуют более высокий уровень обработки и обобщения обнаруженных фактов. Они принимают решения на основе сообщений об обнаруженном подозрительном поведении и явных атаках как от агентов-сенсоров своего хоста, так и от агентов других хостов. *Агенты подавления действий атакующего* (*АПА*) ответственны за

препятствование атакам. *Мета-агент хоста* (*МА*) выполняет обнаружение атак, обрабатывая сообщения от всех агентов хоста в рамках одного хоста, а также участвует в обнаружении сложных распределенных атак, отдельные действия которых направлены на различные хосты сети.

Иерархия планов действий агентов защиты состоит из трех основных уровней обработки:

1) агенты *АС* выполняют первичную обработку сообщений входного трафика;
2) агенты *АИА*, *АРД*, *АОВ* в реальном времени осуществляют предварительный анализ полученных данных, выявляя очевидные атаки;
3) агенты *ИАОВ* и *МА* осуществляют обнаружение многофазных распределенных атак.

Эти агенты также реализуют прогнозирование последующих действий пользователей, используя известные сценарии атак.

4. Прототипы системы моделирования

С использованием средств Java, Visual C++, MASDK и JADE разработаны прототипы отдельных компонентов многоагентной системы моделирования противоборства команд агентов в компьютерных сетях, в частности, прототип симулятора функционирования DDoS-агентов [5], прототип многоагентной системы моделирования DoS-атак [7], и прототип многоагентной системы обнаружения вторжений [7].

Прототип симулятора функционирования DDoS-агентов реализован с использованием JADE и Java. Он позволяет в наглядной форме продемонстрировать все этапы развития DDoS-атак, а также смоделировать различные варианты развития ситуаций в зависимости от заданных параметров защищенности атакуемых сетей. Модель команды агентов представлена в виде трехуровневой структуры, состоящей из «клиента», управляющего командой «мастеров», которые, в свою очередь, контролируют множество «демонов». «Демоны» подразделены на две роли — разведчиков и атакующих, и исполняют фактические действия по нападению на хост-жертву.

Прототипы позволяют в наглядной форме продемонстрировать все этапы развития DDoS-атак, а также смоделировать различные варианты развития ситуаций в зависимости от заданных параметров защищенности атакуемых сетей.

Все проведенные эксперименты с моделированием DDoS-атак были разбиты на два класса:

1) *эксперименты по моделированию атак на макро-уровне*, в которых осуществлялась генерация и исследование атак против модели компьютерной сети;
2) *эксперименты по моделированию атак на микро-уровне* с целью генерация атак против реальной компьютерной сети.

Эксперименты осуществлялись при различных параметрах спецификации задачи атаки и конфигурации атакуемой компьютерной сети. Исследовалось влияние на результативность атак таких параметров, как степень защиты сетевого и персонального межсетевого экрана, степень защиты атакуемого хоста (например, насколько строгим является пароль, имеет ли хост разделяемые файлы, принтеры и другие ресурсы, используются ли доверяемые хосты и др.), уровень знаний злоумышленника о сети и др.

Заключение

В работе предложен подход к многоагентному моделированию противоборства команд программных агентов в сети Интернет.

Подход рассмотрен на примере моделирования процессов реализации распределенных атак «отказ в обслуживании». Приведен фрагмент онтологии предметной области DDoS-атак и механизмов защиты от них. Определена структура команд агентов, механизмы их взаимодействия и координации и спецификация иерархии планов действий агентов. Разработаны прототипы системы моделирования противоборства команд агентов.

Направления будущих исследований связаны с созданием комплексных моделей, архитектуры и программных прототипов для моделирования взаимодействия команды злоумышленников и команды агентов защиты, являющихся полигоном для разработки теоретических основ построения интегрированных многоагентных систем защиты, способных функционировать в антагонистической среде.

Список литературы

[1] *Cohen P. R.*, *Levesque H. J.* Teamwork // Nous. 1991. № 25(4).
[2] *Grosz B.*, *Kraus S.* Collaborative Plans for Complex Group Actions // Artificial Intelligence. 1996. N 86(2).
[3] *Tambe M.*, *Pynadath D. V.* Towards Heterogeneous Agent Teams // Lecture Notes in Artificial Intelligence. V. 2086. Springer Verlag, 2001.
[4] *Kotenko I.* Teamwork of Hackers-Agents: Modeling and Simulation of Coordinated Distributed Attacks on Computer Networks // The 3rd International/Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003). Prague, Czech Republic. Proceedings. Multi-Agent Systems and Applications III // Lecture Notes in Artificial Intelligence. Springer-Verlag, V. 2691. 2003.
[5] *Gorodetski V.*, *Karsayev O.*, *Kotenko I.*, *Khabalov A.* Software Development Kit for Multi-agent Systems Design and Implementation // Lecture Notes in Artificial Intelligence. 2002. V. 2296.
[6] *Котенко И. В.* Многоагентное моделирование атак. Распределенный отказ в обслуживании // КИИ-2004. IX Национальная конференция по искусственному интеллекту с международным участием. Труды конференции. Том 2. М.: Физматлит, 2004.
[7] *Gorodetsky V.*, *Kotenko I.*, *Karsayev O.* The Multi-agent Technologies for Computer Network Security: Attack Simulation, Intrusion Detection and Intrusion Detection Learning // The International Journal of Computer Systems Science & Engineering. 2003. № 4.

Математическая модель безопасного взаимодействия

А. А. Грушо, Е. Е. Тимонина

Рассматрим распределенную компьютерную систему, в которой под узлами будем подразумевать подсистемы компьютерной сети, как локальные сети, так и отдельные компьютеры. Узлы могут обращаться друг к другу за информацией (например, в базу данных), за вычислительными ресурсами или за решением своей задачи. При этом в каждом узле определена своя политика безопасности, имеются свои требования по безопасности и, более того, свое понимание безопасности. Требуется построить модель согласования политик безопасности в такой распределенной системе. При этом модель безопасного взаимодействия узлов в распределенной системе должна быть:

1) простейшей;
2) универсальной;
3) давать основу (язык) для дальнейшего развития безопасного взаимодействия в распределенных системах.

Пусть A и B — узлы распределенной компьютерной системы. Рассмотрим следующую модель безопасного взаимодействия A и B. Пусть A хочет обратиться за информацией (решением задачи) в B. A формирует обращения (задания) к B. Общий вид такого задания имеет следующую форму

$$\{\mathrm{Id}(A)/\ \mathrm{Aut}(A),\ \mathrm{Id}(B),\ \alpha\},$$

где $\mathrm{Id}(A)/\ \mathrm{Aut}(A)$ — идентификатор и аутентификационная информация A, $\mathrm{Id}(B)$ — идентификатор B, α — запрос на информацию или описание задачи для B.

Безопасное взаимодействие подсистем в распределенной системе будем определять через функцию доверия этих подсистем. Доверие будем измерять неотрицательными числами в некоторой линейной шкале. Например, это могут быть действительные числа из отрезка [0, 1]. Пусть для всех упорядоченных пар узлов (A, B) в системе определена функция $D(A, B)$, принимающая неотрицательные значения доверия и

Работа поддержана РФФИ, проект 04-01-00089.

определяющая уровень доверия B к A. Одновременно пусть существует алгоритм, позволяющий для любого задания α, которое может быть выполнено в узле B, вычислить функцию $D(\alpha)$. Значения этой функции также принадлежат множеству уровней доверия. $D(\alpha)$ определяет требуемый уровень доверия к запрашивающему узлу для выполнения задания α в B. Во введенных определениях простейшее правило, определяющее возможность выполнения α в узле B, заключается в следующем. Если $D(A, B) \geqslant D(\alpha)$, то B выполняет задание α. Если $D(A, B) < D(\alpha)$, то к A передается отказ от выполнения задания α.

Вычисление функций $D(\alpha)$ и $D(A, B)$ может проводиться по различным алгоритмам. Для некоторых α вычисления могут быть простыми, а для некоторых α вычисления могут быть не только сложными, но и требующими дополнительной информации. Дополнительная информация может быть получена извне с помощью опросов о параметрах задачи α или доверия к узлу A, или из данных аудита. Значения функции доверия могут меняться с течением времени. Основой для вычисления функции доверия является внутренняя политика безопасности в каждом узле. При этом метод разрешения взаимодействия с помощью функции доверия представляется универсальным для согласования различных политик безопасности. Вообще говоря, каждый узел может по-своему вычислять функции доверия, что может делать ее более открытой или менее открытой для других узлов. Если для собственника узла B нет никаких стимулов решать чужие задачи или делиться информацией, то он может сделать функцию доверия для задач максимальной, а значения функции доверия к остальным узлам минимальными. В этом случае доступ к ресурсам данного узла будет полностью перекрыт.

В системах с повышенными требованиями по безопасности узел A отсылает запрос к доверенной третьей стороне F, через которую осуществляется взаимодействие с B. Функции F состоят в идентификации и аутентификации A, анализе задания и определении возможности выполнения задания α в B.

В системах, где не требуется высокая степень защищенности взаимодействия узлов A и B, запрос указанного вида может быть передан непосредственно узлу B. Если предположить наличие враждебных для B узлов, то при непосредственном взаимодействии возможны атаки против B.

Возможности взаимодействия в распределенной системе описываются ориентированным графом G, вершины которого — узлы распределенной сети, дуги определяются возможностями обращения одних узлов к другим и помечены значениями функции доверия для соответствующих пар узлов.

Если в графе G некоторые узлы связаны ориентированным маршрутом, то, вообще говоря, возможна передача запроса на выполнение задания следующему по маршруту узлу графа. Такую передачу задания будем называть транзитом задания.

Мы предложили некоторый способ безопасного взаимодействия узлов распределенной сети. Является ли данный способ непротиворечивым? В каких условиях он является непротиворечивым?

1. В случае наличия доверенного центра политика, определяемая функцией доверия, непротиворечива, так как F не допускает транзита заданий.
2. Если доверенный центр отсутствует и допустим транзит заданий, то политика безопасности может быть противоречива.

Приведем примеры некоторых функций доверия для узла B. Будем считать, что доверие измеряется действительными числами из отрезка [0, 1]. Определим $U(\alpha)$ — потенциальный ущерб для активов узла B от выполнения задания α, и U — сумма всех активов данного узла. Тогда отношение $U(\alpha)/U$ — относительная величина потенциального ущерба от выполнения задания α. Функция доверия $D(\alpha)$ в этом случае, естественно, определяется следующей формулой

$$D(\alpha) = \frac{U(\alpha)}{U}.$$

Для определения функции $D(A, B)$ может использоваться следующая информация:

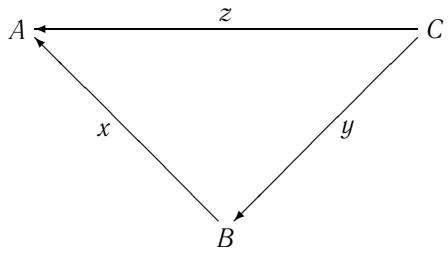
- безупречность истории взаимодействия A и B;
- поручительство других организаций за A;
- данные аудита по накопленным опасным событиям, связанным с запросами от A;
- согласование функций доверия для различных узлов в случае, когда допустимы транзитные передачи заданий и запросов другим узлам;
- случаи ущерба в узле B.

Вычисление функции $D(A, B)$ на узле B может быть произвольным и определяться внутренней политикой безопасности в B.

Рассмотрим некоторые принципы изменения значений функции доверия. В случае, когда доверие принимает значения на отрезке [0, 1], значения функции доверия могут рассматриваться как принадлежность к нечеткому множеству доверенных клиентов. Тогда, естественно, при наступлении различных событий правила принадлежности к нечеткому множеству определяются с помощью одного из вариантов нечеткой логики, которые исследовались в научной и практической литературе.

Такой подход позволяет обеспечить быстрое и адаптивное вычисление новых значений функции доверия в зависимости от истории взаимодействия с различными узлами и особенностей внутренней политики безопасности.

Пример. Рассмотрим ограничения на функцию доверия в тех случаях, когда допускается транзит информации и заданий через данный узел в другие узлы. Пример. Рассмотрим три узла A, B и C и построим схему взаимодействия, указанную стрелками на рисунке, которая позволит объяснить сущность возникающих ограничений.



Пусть $x = D(B, A)$, $y = D(C, B)$ и $z = D(C, A)$. Пусть A не верит C, так что $z < x$, или $z < y$. Если A представляет собой базу данных и α связано с получением информации из A, то если A верит B так, что готов выполнить запрос α и B верит C так, что готов выполнить α, то транзит информации от A через B к C равносилен выполнению задания α для C от A, что противоречит предположению, что A не верит C. Отсюда следует следующее ограничение

$$z \geqslant \min(x, y).$$

Эти ограничения естественно обобщаются на произвольную пару ориентированных цепей, ведущих от C к A в графе G, порожденном узлами распределенной системы и ориентированными дугами, помеченными функциями доверия:

$$G = (A, B, C, \ldots, D(A, B)).$$

Лемма 1. *Пусть*

$$C, B_1, B_2, \ldots, B_n, A$$

и

$$C, E_1, E_2, \ldots, E_m, A$$

— два маршрута из C в A. И пусть во всей распределенной сети функция доверия к любому заданию в каждом узле одинакова и

допускается транзит заданий по узлам распределенной системы. Предположим, что функция доверия принимает значения на отрезке [0, 1] *и для любого значения* $x \in [0, 1]$ *может существовать задание* α *такое, что* $D(\alpha) = x$. *Тогда ограничения на функцию попарного доверия определяются следующими равенствами*

$$\min(D(C, B_1), D(B_1, B_2), \ldots, D(B_n, A)) = \\ = \min(D(C, E_1), D(E_1, E_2), \ldots, D(E_m, A)).$$

Доказательство. Пусть задание α для A имеет доверие $D(\alpha)$. Тогда если

$$\min(D(C, B_1), D(B_1, B_2), \ldots, D(B_n, A)) = D(\alpha),$$

то C может получить результат вычисления задания α для A по маршруту

$$C, B_1, B_2, \ldots, B_n, A.$$

Если

$$\min(D(C, E_1), D(E_1, E_2), \ldots, D(E_m, A)) < D(\alpha),$$

то C не получит результат вычисления задания α для A по маршруту

$$C, E_1, E_2, \ldots, E_m, A.$$

Это означает противоречивость политики безопасности. Тогда

$$\min(D(C, B_1), D(B_1, B_2), \ldots, D(B_n, A)) \leqslant \\ \leqslant \min(D(C, E_1), D(E_1, E_2), \ldots, D(E_m, A)).$$

Из условия леммы 1 аналогично получаем обратное неравенство. Лемма доказана. □

Таким образом, непротиворечивость политики безопасности в распределенной системе, в которой допускается транзит заданий между узлами, связана с необходимостью выполнения ограничений по всем ориентированным маршрутам между любой парой вершин. Проверка этих ограничений представляется трудной задачей, хотя для небольших графов эта задача разрешима.

Рассмотрим дополнительные ограничения, связанные с симметрией функции доверия в графе G. То есть предположим что если $D(A, B) = x$, то $D(B, A) = x$ для всех A, B. Пусть по-прежнему во всей распределенной сети функция доверия к любому заданию в каждом узле одинакова и допускается транзит заданий по узлам распределенной системы. Предположим, что функция доверия принимает значения на отрезке [0, 1] и для любого значения $x \in [0, 1]$ может существовать задание α такое, что $D(\alpha) = x$. Тогда граф G становится неориентированным и справедлива следующая лемма.

Лемма 2. *Функция* $D(A, B)$ *постоянна на любом цикле компоненты связности графа* G.

Доказательство. В силу сделанных предположений для графа G выполняются условия леммы 1. Тогда для любого цикла в компоненте связности графа G доверие любого ребра равно минимуму значений функции доверия по остальным ребрам цикла. Это значит, что значения функции доверия для любого ребра на цикле больше, либо равняется значению функции доверия любого другого ребра на этом цикле. Отсюда следует, что все значения функции доверия на цикле одинаковы. Лемма доказана. □

Теорема. *Если компонента связности графа* G *является двусвязной, то значения функции доверия постоянны на всех ребрах этой компоненты.*

Доказательство. Из двусвязности графа G следует, что любые два ребра графа G лежат на простом цикле [1]. Отсюда и из леммы 2 следует результат теоремы. □

Следствие. *Если в распределенной системе каждый узел может обращаться к каждому узлу с заданием или запросом на информацию, допускается транзит заданий, функция доверия симметрична для каждой пары узлов, а значения функции доверия к заданию одинаковы на всех узлах и принимают произвольные значения от* 0 *до* 1, *то функция доверия для пар узлов является константой.*

Доказательство. Обращение любого узла к любому узлу означает, что граф G является полным, а, следовательно, двусвязным. Тогда из теоремы следует следствие. □

Рассмотренная в следствии ситуация означает одинаковое доверие к друг другу всех узлов, а все задания распадаются на два класса. Задания, которые можно выполнять только внутри любого из узлов, и задания, с которыми можно обращаться к любому другому узлу. Задания разделяются по уровню доверия. Если уровень доверия задания больше, чем единый уровень доверия к соседям, то такое задание может выполняться только внутри узла. Если уровень доверия к заданию меньше, чем общий уровень доверия к соседям, то такое задание может выполнять в любом другом узле.

Выводы.

1. Согласование политик безопасности с помощью функции доверия является эффективным, если в системе недопустим транзит информации и заданий. Например, в случае, когда топология распределенной системы имеет вид звезды, где допустимость выполнения заданий определяется в доверенном центре F (в центре звезды).

2. В случае произвольного графа G и допустимости транзита заданий согласование политик безопасности с помощью функции доверия в общем случае становится не эффективным.

3. В случае симметричной функции доверия между парами узлов согласование политик безопасности с помощью функции доверия может быть эффективным, если граф является двусвязным.

Список литературы

[1] *Емеличев В. А.*, *Мельников О. И.*, *Сарванов В. И.*, *Тышкевич Р. И.* Лекции по теории графов. М.: Наука, 1990, 384 с.

Расширенная модель безопасного взаимодействия в распределенных системах

М. В. Большаков

В данной работе рассматриваются свойства расширенной простой модели согласования политик безопасности, описанной в настоящем сборнике [1] и предложенной для поддержки взаимодействия подсистем или виртуальных организаций в системах, построенных на основе технологий GRID.

1. Модель согласования политик безопасности

Модель предложенная А. А. Грушо и Е. В. Тимониной и представленная в настоящем сборнике [1] заключается в следующем[1]. Пусть A и B принадлежат множеству $\mathcal{C}$ взаимодействующих подсистем в рамках одной виртуальной организации или виртуальных организаций в составе большой, объединяющей их GRID структуры. Пусть B хочет обратиться за информацией (решением задачи) в A. Для этого B формирует обращение (задание) к A. Общий вид такого задания имеет следующую форму

$$\{\mathrm{Id}(B)/\mathrm{Aut}(B),\ \mathrm{Id}(A),\ \alpha(\mathrm{Id}(B))\},$$

где $\mathrm{Id}(B)/\mathrm{Aut}(B)$ — идентификатор и аутентификационная информация B, $\mathrm{Id}(A)$ — идентификатор A, $\alpha(\mathrm{Id}(B))$ — запрос на информацию или описание задачи для A.

Безопасное взаимодействие подсистем в распределенной системе будем определять через функцию доверия этих подсистем. Доверие будем измерять неотрицательными числами в некоторой линейной шкале $\mathcal{L}$, которую можно описать следующим образом.

- Для всех A, $B \in \mathcal{C}$ определена функция $DC_A(B)\colon \mathcal{C} \to \mathcal{L}$ описывающая уровень доверия A к B, и $\forall\ B \in \mathcal{C}$.
- Для всех $A \in \mathcal{C}$ и задачи $\alpha \in \mathcal{T}$ определена функция $DT_A(\alpha)\colon \mathcal{T} \to \mathcal{L}$, описывающая необходимый уровень доверия для исполнения задачи α на узле A.

[1] Обозначения функций в данной статье несколько отличны от использованных авторами [1].

Таким образом, условие исполнения задачи α из узла B в узле A естественно записать в виде $DC_A(B) \geqslant DT_A(\alpha(\mathrm{Id}(B))$, в противном случае, A передает B отказ в исполнении. Как было замечено в [1] вычисление функций $DC_A(B)$ и $DT_A(\alpha(\mathrm{Id}(B))$ может проводиться по различным алгоритмам. Для некоторых вычисления могут быть простыми, а для некоторых вычисления могут быть не только сложными, но и требующими дополнительной информации. Дополнительная информация может быть получена извне с помощью опросов о параметрах задачи или доверия к узлу B, или из данных аудита. Значения функции доверия могут меняться с течением времени. Основой для вычисления функции доверия является внутренняя политика безопасности в каждом узле или в каждой виртуальной организации.

2. Методы транзита заданий

В [1] были рассмотрены принципы формирования функций доверия и описаны условия корректности задания этих функций при условии отсутствия прямого транзита заданий. Под транзитом понималась следующая схема передачи задач: компоненты задания соответствующие аутентификации и идентификации отправителя и получателя изменялись, что соответствует режиму изменения полномочий. В данной работе рассмотрим другой метод транзита задач, который будем называть пересылкой (то есть, задание по пути следования изменяться не будет).

1. Транзитом с изменением полномочий — будем называть следующее преобразование задания. Узел осуществляющий транзит подставляет в качестве идентификатора, аутентификатора свои собственные идентификатор и аутентификаторы, а в качестве исполняющего узла — следующий по пути следования к конечному.
2. Пересылка заданий — транзит, при котором, задание пересылается на следующий узел, без каких-либо изменений, при этом, возможно, факт пересылки запоминается, для предотвращения возникновения циклов.

В общем случае, первый метод транзита позволяет изменять и само описание задачи, переформулируя задачу безопасным образом, поэтому уровень доверия при транзите с изменением полномочий может удовлетворять следующему соотношению: $DC_A(C) > DC_A(B)$ при транзите задания $B \Longrightarrow C \Longrightarrow A$, то есть узел B может исполнять некоторое подмножество задач недоступных ему при использовании пересылки, или прямом обращении к узлу A.

С другой стороны, использование пересылки позволяет не тратить узлу C время на преобразование задачи и предоставить прямой доступ к

узлам, на которые в связи с особенностями сетевой структуры пересылать задания напрямую нельзя. Стоит отметить, что первый метод транзита обеспечивает анонимность оригинального отправителя задания при соответствующем преобразовании задания (когда никакая информация об оригинальном отправителе не попадает в описание сформированной задачи).

Для корректного описания пересылки заданий добавим к модели следующую сетевую структуру:

- $G = (\mathcal{C}, E)$ — ориентированный граф на множестве узлов, описывающий возможные пути передачи заданий.

3. «Безопасная» пересылка заданий

Далее приведем ряд утверждений, связанных с естественным определением «безопасной» пересылки заданий.

Пусть задание α последовательно пересылается по пути $P = \langle p_i \rangle$, $i = 1, \ldots, n$ через вершины p_i, где $p_1 = B$, $p_n = A$, тогда можно ввести следующие определения.

Определение 1. Пересылку задания из узла B в узел A по пути P будем называть безопасной, если $\forall\ p_i,\ p_j \in P$ вершины $DC_A(p_i) \leqslant \leqslant DC_A(p_j)$ в случае $i \leqslant j$.

Определение 2. Подграф G_A графа G будем называть опирающимся на вершину A, если $A \in G_A$ и $\forall$ вершины $B \in G_A$ существует путь из B в A в G_A и из A ребер не выходит.

Определение 3. Опирающийся подграф G_A графа G будем называть максимальным если для любой вершины $C \in G$ и $C \notin G_A$ не существует пути из C в A в G и для всех ребер $r = (D, B)$, $D \in (G_A \setminus A)$, $B \in G_A$ если $r \in G$, то и $r \in G_A$.

На основании данных определений можно сформулировать и доказать следующую теорему.

Теорема 1 (О безопасных путях в опирающемся подграфе). *Пусть выбран опирающийся на вершину A подграф G_A графа G, задания могут передаваться только по ребрам G_A и $\forall\ l \in L\ \exists\ \alpha$ такое, что $DT_A(\alpha) = l$. При этих условиях пересылки по любым путям в G_A безопасны $\Longleftrightarrow \forall\ r = (D, B)$, $D \in (G_A \setminus A)$, $B \in G_A$, $DC_A(D) \leqslant DC_A(B)$, то есть, — все ребра ведут от менее доверенных вершин к более доверенным.*

Доказательство. Необходимость, очевидно, следует из того, что DC_A по всем ребрам не убывает.

Докажем далее следствие. Предположим противное, а именно: $\exists\ r = (D, B)$, $D \in (G_A \setminus A)$, $B \in G_A$ такое что $DC_A(D) > DC_A(B)$. Тогда

рассмотрим путь из D в A проходящий по этому ребру через B. Такой путь существует, так как существует путь из B в A. По условиям теоремы существует α такая, что $DC_A(D) \geqslant DT_A(\alpha) > DC_A(B)$. На этом основании, пересылка задания $\{\mathrm{Id}(D)/\ \mathrm{Aut}(D),\ \mathrm{Id}(A),\ \alpha(\mathrm{Id}(D))\}$ по пути через узел B не будет безопасной, что приводит к противоречию. □

Следствие. *В условиях теоремы* 1 DC_A *постоянна на любом цикле в* G_A.

Доказательство. Доказательство следует из записи неравенств для ребер цикла. □

Различные модели маршрутизации заданий могут порождать различные опирающиеся подграфы.

Пример 1. Модель маршрутизации заданий в A «поиском в глубину из исходной вершины при случайном выборе первого соседа» соответствует максимальному, опирающемуся на A подграфу.

Заметим, что при такой маршрутизации все вершины модели «звезда» (в которой выделен центр и все задания могут «ходить» только через него) обязаны иметь постоянную функцию DC, так как есть цикл между центром и каждой вершиной.

Пример 2. Модель маршрутизации заданий в A «по кратчайшему пути» соответствует опирающемуся на A подграфу, получившемуся в результате построения остовного подграфа (для максимального опирающегося на A подграфа) поиском из A в ширину.

В этом случае, функция DC не обязана быть тривиальной. Очевидно только, что для центра она должна быть максимальна.

Замечание. Практически, никогда нельзя гарантировать соблюдения злоумышленником, захватившим узел, правил маршрутизации, используемых в данной сети. С учетом этого факта, анализируя безопасность, следует рассматривать максимальное опирающееся на вершину дерево.

4. Оценка урона системе при захвате GRID нескольких узлов

При разработке конфигураций различных систем GRID, необходимо оценивать полученные конфигурации на предмет стойкости к различным воздействиям на узлы системы. В графовых структурах такими оценками являются числа ω и λ графа.

- λ — реберная устойчивость графа, минимальное количество ребер, при удалении которых граф становится не связным.
- ω — вершинная устойчивость графа, минимальное количество вершин, при удалении которых граф становится не связным.

Для GRID-структур, заданных графом $G = (\mathcal{C}, E)$ возможных взаимодействий и функциями доверия $DC_A(B)$, $DT_A(\alpha(\mathrm{Id}(B))$ где $A, B \in \mathcal{C}$, будем рассматривать функцию $U(G, DC, l)$, $l \in \mathcal{L}$ равную минимальному количеству узлов в сети захваченных противником так, что на любом из узлов сети противник с помощью транзита или пересылки заданий может выполнять задание с уровнем привилегии большим или равным l:

$$U(G, DC, l) = \min_{|C|,\ C \subseteq \mathcal{C}} \{\forall\ a \in \mathcal{C},\ \exists\ c \in C : DC_v(c) \geqslant l\}.$$

Данная функция позволяет представить интегральную характеристику возможного урона, нанесенного GRID-системе, в результате точечных воздействий. Очевидно, что:

- функция U нигде не убывает на $l \in [l_{\min}, l_{\max}]$,
- $U(G, DC, l_{\min}) \geqslant 1$,
- $U(G, DC, l_{\max}) \leqslant |\mathcal{C}|$.

С точки зрения анализа уязвимости представляет интерес поведение функции в промежутке $[l_{\min}, l_{\max}]$. Приведем несколько эмпирических утверждений о свойствах хорошей с точки зрения такого поведения функции U.

1. Функция U должна достигать своего максимума как можно раньше. В лучшем случае — $U(G, DC, l_{\min}) = |\mathcal{C}|$, что означает отсутствие передачи прав другим узлам.
2. Рост функции U должен быть как можно более быстрым, в идеале — U это скачок с 1 на $|\mathcal{C}|$ по достижении некоторого порога. Например, администрировать узлы безопаснее всего локально.
3. Рост функции U должен быть в каком-то смысле «равномерным». Большие промежутки, на которых функция U постоянна могут сигнализировать о потенциальной ошибке в структуре GRID-системы, а именно, о наличии узлов, имеющих привилегированный доступ на большое количество других участников.

Следует заметить, что перечисленные условия бесполезны с точки зрения практических задач, так как они переформулируют очевидное утверждение, что самый защищенный в сети компьютер — компьютер, отключенный от сети. Однако, поведение функции может показать наличие узлов уязвимости, защиту которых нужно планировать и проверять более тщательно.

Определение 4. Графом доверия уровня l, G_l, будем называть граф, получаемый из графа связей участников GRID выкидыванием связей, доверие по которым меньше l.

Функцию $U(G, DC, l)$ предлагаем строить поточечно на промежутках постоянства G_l. Упорядочивая $DC_A(B) = a_i$, $i \in [1, n^2]$ для всех

$A, B \in \mathcal{C}$, получим последовательность $\{a_j\}$, $j \in [1, n^2]$, где j — некоторая перестановка индексов i и $a_j \leqslant a_{j+1}$. Очевидно, что для всех $l \in [a_j, a_{j+1})$ граф G_l не изменяется, поэтому достаточно посчитать $U(G, DC, l)$ лишь в одной точке из указанного промежутка.

При фиксированном l вычисление функции $U(G, DC, l)$ представляет собой поиск минимального покрытия в G_l. К сожалению, не известен ни один полиномиальный алгоритм для поиска минимального покрытия для случая произвольных графов (см. [2, с. 167]), таким образом, на больших размерностях вычисление неэффективно.

Заключение

Дальнейшие исследования предложенной модели предполагается вести в следующих направлениях.

- Пересылку задания можно рассматривать как, может, простую, но также требующую ресурсов задачу, то есть, имеющую собственный уровень доступа.
- Интерес представляет разработка быстрых методов минимизации функции урона и методов оптимального построения GRID систем.

Список литературы

[1] *Грушо А. А.*, *Тимонина Е. Е.* Математическая модель безопасного взаимодействия. Представлена в настоящем сборнике.

[2] *Липский В.* Комбинаторика для программистов. М.: Мир, 1988.

Исчисление событий для спецификации и верификации политик безопасности защищенной вычислительной сети

А. В. Тишков, И. В. Котенко

Введение

В настоящее время в Санкт-Петербургском институте информатики и автоматизации разрабатывается система верификации средств защиты вычислительной сети (СВЕРЗ) с применением гибридного подхода, основанного на методах теории доказательств, методах проверки на модели и методах имитационного моделирования.

В данной работе рассматривается использование методов теории доказательств для спецификации и верификации политик безопасности.

Предлагаемый подход, основанный на применении теории доказательств, базируется на использовании исчисления событий, впервые введенного в [1], и нашедшего множество практических применений, в том числе в задачах информационной безопасности [2].

Исчисление событий реализует обобщенный принцип инерции «система остается неизменной пока ее состояние не изменит то или иное событие». Исчисление событий имеет множество расширений и интерпретаций [3]. В работе предлагается один из базовых вариантов исчисления событий.

Определение событий и действий в исчислении событий является семантически близким к событийно управляемой объектно-ориентированной парадигме программирования. Это обстоятельство позволяет осуществить реализацию системы верификации политик безопасности на объектно-ориентированном языке высокого уровня с использованием средств логического программирования.

Используемая модель безопасности компьютерной сети состоит из формального описания защищаемой вычислительной сети и политик

Работа выполняется при поддержке РФФИ (проект 04-01-00167), программы фундаментальных исследований ОИТВС РАН (контракт 3.2/03) и программы FP6 Европейского Сообщества.

безопасности. Язык описания вычислительной сети определяет набор элементов сети, содержащий как технические, так и программные средства, доступ к которым необходимо регулировать с помощью политик. Язык задания политик безопасности предназначен для определения ограничений, при которых субъекты имеют доступ к объектам.

Исчисление событий используется для задания поведения системы, верификации политик, заданных администратором, и верификации уточненных политик, являющихся результатом применения исходных политик безопасности к описанию конкретной компьютерной системы.

1. Основные элементы исчисления событий

Исчисление событий представляет собой логику первого порядка, расширенную тремя сортами:

1) *действия* — свойства окружающего мира, начинающиеся и завершающиеся благодаря возникновению *событий*,
2) события начинают или завершают действия,
3) время — целые или вещественные неотрицательные числа.

1.1. Базовые предикаты

Наступление события, начало и завершение действия выражаются предикатами Happens, Terminates и Initiates соответственно.

- Happens(a, t) принимает значение «истина», если событие a возникает в момент времени t.
- Initiates(a, f, t) принимает значение «истина», если событие a происходит в момент времени t, а f действует после t (но не в t).
- Terminates(a, f, t) принимает значение «истина», если событие a происходит в момент времени t, а действие f не продолжается после t (но f все еще действует в момент t).

Предикат HoldsAt(f, t) принимает значение «истина», если f действует в момент времени t. Предикаты InitiallyTrue(f) и InitiallyFalse(f) определяют действует ли f в начальный момент времени 0.

1.2. Аксиомы ИС

Вспомогательный предикат Clipped(t_1, f, t_2) принимает значение «истина», если действие f было завершено в течение промежутка времени [$t1$, $t2$). Первая аксиома ИС связывает этот предикат с Happens и Terminates:

$$\text{Clipped}(t_1, f, t_2) \leftarrow \text{Happens}(a, t_1) \wedge t_1 \leqslant t < t_2 \wedge \text{Terminates}(f, t_2).$$

Вспомогательный предикат Declipped принимает значение «истина» если действие f было начато в промежутке времени [t_1, t_2). Вторая аксиома ИС связывает этот предикат с Happens и Initiates:

$$\text{Declipped}(t_1, f, t_2) \leftarrow \text{Happens}(a, t_1) \wedge t_1 \leqslant t < t_2 \wedge \text{Initiates}(f, t_2).$$

Следующие четыре аксиомы позволяют определить состояние действия, в начальный и последующие моменты времени:

$$\begin{aligned}
\text{HoldsAt}(f, t_2) &\leftarrow \text{Happens}(a, t_1) \wedge \text{Initiates}(a, f, t_1) \wedge \\
&\qquad \wedge t_1 < t_2 \wedge \neg\, \text{Clipped}(t_1, f, t_2), \\
\neg\, \text{HoldsAt}(f, t_2) &\leftarrow \text{Happens}(a, t_1) \wedge \text{Terminates}(a, f, t_1) \wedge \\
&\qquad \wedge t_1 < t_2 \wedge \neg\, \text{Declipped}(t_1, f, t_2), \\
\text{HoldsAt}(f, t) &\leftarrow \text{InitiallyTrue}(f) \wedge \neg\, \text{Clipped}(0, f, t), \\
\neg\, \text{HoldsAt}(f, t) &\leftarrow \text{InitiallyFalse}(f) \wedge \neg\, \text{Declipped}(0, f, t), \\
\text{InitiallyTrue}(f) &\vee \text{InitiallyFalse}(f).
\end{aligned}$$

2. Реализация аксиоматики на ILOG JRules

Механизм вывода в ILOG JRules [4] реализует спецификацию [5], что позволяет использовать его API в Java-приложениях. База фактов располагается в рабочей области, в которую при помощи инициализирующих правил заносятся объекты. В предлагаемой реализации каждый объект представляет собой экземпляр одного из Java-классов, относящихся к аксиоматике ИС. Правило ILOG JRules состоит из двух частей: условия (сравнения по образцу) и действия, изменяющего состояние рабочей области.

Реализация построена таким образом, что для каждого истинного в конкретный момент времени предиката, соответствующий ему Java-объект находится в рабочей области JRules. Пример реализации первой аксиомы ИС:

```
when {
  ECTimePoint(?t1: time);
  ECTimePoint(?t2: time; time > ?t1);
  ECHappens(?e: eventName; ?t: time; time > ?t1; time <= ?t2);
  ECFluent(?f: name);
  ECTerminates(eventName equals ?e; fluent equals ?f; time == ?t);
  not ECClipped(fluent == ?f; time1 == ?t1; time2 == ?t2);
}
then {
  assert ECClipped
  {
```

```
    fluent = ?f;
    time1 = ?t1;
    time2 = ?t2;
  }
}
```

В настоящем примере переменные `t1` и `t2` используются для задания промежутка времени возникновения события, прерывающего действие `f`. Третья и четвертая строки листинга задают границы промежутка как (`t1`, `t2`]. В четвертой–шестой строках рабочая область проверяется на наличие объектов, соответствующих возникновению события `e`, завершающего действие `f`. Наконец, правая часть правила (`then`) добавляет в рабочую область объект `ECClipped` в том случае, если такого объекта в ней не было (строка 7).

3. Приложения

Приведенная реализация исчисления событий применяется для описания управления политиками безопасности для открытых и закрытых систем.

В открытой системе ограничения накладываются на субъекты, тогда как объекты не защищены. В этом случае используются политики облигации (обязательного выполнения действия над объектом) и запрета (отказа субъекту в доступе к объекту).

Реализация управления возлагается на программное обеспечение агента субъекта, который либо принимает запрос на доступ как обязательный к выполнению, либо запрос отклоняется.

В закрытой системе используются политики положительной и отрицательной авторизации доступа к субъекту, за реализацию которых отвечает контроллер объекта. В общем случае система безопасности содержит оба указанных программных компонента, при этом агент субъекта, в случае допуска субъекта к объекту запрашивает авторизацию у контроллера субъекта.

При реализации описанной системы безопасности запросы субъекта на доступ к ресурсу, а также запросы агента субъекта к контроллеру объекта выражаются в виде событий исчисления событий, а операции предоставления или отклонения доступа — в виде действий.

Другое применение исчисления событий заключается в реализации модели ролевого доступа RBAC. Один из четырех основных компонентов RBAC, динамическое распределение обязанностей, заключается в авторизации пользовательской сессии на новую роль во время сеанса работы с приложениями защищенной сети. Запрос пользователя на доступ к приложению, требующему использования активации новой роли может рассматриваться как событие в исчислении событий, в то время как активное состояние той или иной роли, приписанной пользователю представляет собой действие в исчислении событий. Решаемая задача заключается в исследовании, какая роль может быть активна для пользователя в заданный момент времени при имеющихся ограничениях, задаваемых через правила динамического распределения обязанностей.

4. Заключение

В работе рассмотрено использование для спецификации и верификации политик безопасности методов теории доказательств, базирующегося на использовании специального исчисления событий.

Предложена реализация аксиоматики исчисления событий на ILOG JRules.

Данная аксиоматика была опробована на примере моделирования работы пользователей в компьютерном классе исследовательской группы информационных технологий в образовании СПИИРАН.

Дальнейшие исследования связаны с интергацией приложения на базе ILOG JRules со средствами проверки на модели SPIN, NuSMV.

Список литературы

[1] *Kowalski R. A.*, *Sergot M. J.* A Logic-Based Calculus of Events. New Generation Computing, 4: 67–95, 1986.

[2] *Bandara A.*, *Lupu E.*, *Russo A.* Using Event Calculus to Formalize Policy Specifications and Analysis. In IEEE 4th International Workshop on Policies for Distributed Systems and Networks, pages 26–39, June 2003.

[3] *Miller R*, *Shanahan M.* The Event Calculus in Classical Logic. Linkoping Electronic Articles in Computer and Information Science, 4(16), 1999.

[4] ILOG JRules. `http://www.ilog.com/products/jrules/`.

[5] JSR-000094 JavaTM Rule Engine API. `http://www.jcp.org/aboutJava/communityprocess/review/jsr094/`.

Сравнение ролевой и дискреционной моделей разграничения доступа

О. О. Андреев

В данной работе рассматриваются две распространенные модели разграничения доступа — дискреционная (DAC) и ролевая (RBAC). Сравнивается функциональность этих моделей и исследуется вопрос выражения формально-логического представления контроля доступа в одной модели через ее представление в другой.

Введение

В настоящее время в связи с бурным развитием вычислительных средств и сетей передачи данных, увеличением числа пользователей этих систем все большее внимание уделяется вопросам контроля доступа к их ресурсам. Подобный контроль реализуется через заданные администратором ограничения доступа, то есть, возможности выполнить определенные действия, пользователей по отношению к ресурсам системы. Для этого в определенной математической модели задается формальный, строго определенный набор правил. Эти правила, как правило, записываются на некотором языке. Существуют различные логико-языковые средства описания контроля доступа, среди которых одним из самых известных является OASIS Standard eXtensible Access Control Markup Language (XACML), созданный консорциумом OASIS Open [6]. В данной работе рассматриваются только математические модели, не затрагивая вопросы реализации логико-языковых и собственно программных средств контроля доступа.

В настоящее время существуют различные модели контроля доступа. Самыми распространенными являются мандатная (MAC), дискреционная (DAC) и ролевая (RBAC). Первые две модели были созданы еще в 70-е годы и используются в большинстве эксплуатируемых компьютерных систем. Ролевая модель появилась сравнительно недавно и на настоящий момент распространена не столь широко, хотя, как будет показано далее, в связи с большей приспособленностью этой модели к

крупным информационно-вычислительным комплексам с большим количеством пользователей, данная модель привлекает все больший интерес. В настоящей работе будут рассмотрены только дискреционная и ролевая модели, так как мандатная модель имеет определенную специфику применения, как правило, в организациях, в которых главным требованием по безопасности является неразглашение тайны.

Одной из важных в распределенных сетевых структурах является задача переноса политик безопасности из одной системы в другую и одним из первым, возникающих в этой связи вопросов, является вопрос о возможности переноса модели разграничения доступа, записанной (сформулированной) для системы с одной политикой безопасности в другую, с отличной от первой политикой и моделью разграничения доступа. В данной статье этот вопрос рассматривается для двух упомянутых выше моделей.

1. Дискреционная модель

Как уже было отмечено ранее, дискреционная модель появилась еще в 70-х годах и завоевала с того времени широкую популярность, в том числе, — своей простотой. В настоящее время большинство популярных операционных систем (операционные системы — ОС Linux, Solaris, Windows 2000) поддерживают контроль доступа на основе дискреционной модели. Реализация именно этой модели является одним из требований, предъявляемых к защищенным системам по критерию Department of Defense Trusted Computer System Evaluation Criteria [3], [4].

В основе этой модели лежат три основных понятия: *субъекты*, *объекты* и *доступы*. Например, субъектами могут являться пользователи, объектами — файлы, а доступами — чтение, запись и исполнение. Контроль доступа осуществляется при помощи *матрицы доступа*, в которой для каждого доступа каждого субъекта к каждому объекту указывается, разрешен он или запрещен. Помимо этого в большинстве систем введены два особых доступа: *владелец* и *владелец с правом передачи* (в некоторых системах не запрещается существование у объекта нескольких владельцев, однако существование хотя бы одного владельца в большинстве случаев является обязательным). Пользователь, являющийся владельцем объекта, может изменять доступы других пользователей к нему, за исключением создания или удаления владельцев. Если же он является владельцем с правом передачи, то он может изменить любой доступ. Существующие реализации зачастую поддерживают лишь часть этой функциональности, например, любой владелец

имеет право передачи (и называется владельцем). В ОС UNIX также есть «суперпользователь» (root), для которого разрешены все доступы ко всем объектам.

Дадим формальное определение вышеизложенной модели.

Определение 1. Дискреционной моделью называется:

- S, множество пользователей;
- O, множество объектов;
- A, множество доступов;
- $M \subseteq S \times O \times A$, матрица доступа (элементами матрицы являются подмножества множества доступов A).

С учетом данного определения разрешается доступ a субъекта s к объекту o, если $(s, o, a) \in M$. Если $(s, o, c) \in M$, то субъект может добавлять или удалять из M тройки вида (t, o, a) для любых $t \in S$ и $a \in A \setminus \{c, ct\}$. Если $(s, o, ct) \in M$, то этот субъект может добавлять и доступы c и ct. Таким образом, если у субъекта есть оба доступа «владелец» и «владелец с правом передачи», то последний доступ является опеределяющим. Как уже было отмечено, некоторые системы вводят дополнительные условия, например, существование одного и только одного владельца у каждого объекта: $\forall\, o\ \exists!\, s\colon (s, o, c) \in M$ (здесь и далее $\exists!$ — существует и единственный).

Одним из важнейших преимуществ и причин столь широкой распространенности дискреционной модели, помимо ее простоты, является то обстоятельство, что администратор (и владельцы объектов) может точно контролировать любой доступ (в отличие от мандатной модели, где доступ предоставляется сразу всем пользователям определенного уровня доступа). Однако, платой за эту точность является увеличение размера матрицы доступа при увеличении числа пользователей, а главное, усложнение работы и контроля за ней. Например, владельцу объекта необходимо устанавливать множество доступов к этому объекту для всех пользователей в системе, при изменении статуса владельца необходимо проверить и переустановить для него доступы ко всем объектам.

2. Ролевая модель

Как уже отмечалось в предыдущей части, при увеличении числа пользователей системы или сети до нескольких сотен и тысяч (вполне типичная ситуация для крупных корпоративных сетей), управление матрицей доступа, установка доступов для новых объектов и управление доступом к уже имеющимся становится весьма обременительной задачей. Одной из целей, которые ставили перед собой разработчики ролевой модели являлось упрощение процесса администрирования системы.

В последнее время ролевая модель получает все большее распространение. Она присутствует в RSBAC Linux (надстройке ядра и пакете программ для ОС Linux, которые поддерживают различные модели контроля доступа, где RBAC — в числе главных), а также ОС Solaris 8 и выше [5]. Необходимо отметить также очень известный механизм `sudo`, который существует почти на всех UNIX-подобных операционных системах, который также имеет частичное сходство с ролевой моделью (сравнение `sudo` с приводится в [5]). Однако, и это следует отметить, одним из его недостатков является недостаточная гранулярность выдачи привилегий, так как этот механизм позволяет запускать с повышенным уровнем доступа программы целиком. Интерес к этим средствам вызван тем, что для выполнения большого числа операций во многих современных операционных системах необходим уровень доступа *root*, который имеет права на любые действия (в том числе возможность чтения/записи любых файлов), образуя таким образом модель «все или ничего».

Ролевая модель базируется на понятиях *пользователей*, *ролей* и *привилегий*. В отличие от дискреционной модели здесь отсутствуют понятия объекта и доступа к нему, оба этих понятия заменяются понятием привилегии, то есть, возможности выполнить какое-то действие (например, прочесть некоторые данные, запустить приложение). Понятие владельца информации (владельца объекта) в данном случае не может существовать. Подразумевается, что вся информация, которую создают пользователи, принадлежит организации, в которой они работают. Каждый пользователь может (и, более того, должен) принадлежать к одной или нескольким ролям, которые характеризют его положение и должность в организации, которой принадлежит система (одним из примеров является больница: ролями в ней могут быть медсестра, хирург, фармацевт и т. д.) Привилегии назначаются не напрямую пользователям, а ролям (например, привилегия «доступ к истории болезни больного» может быть дана роли «лечащий врач», а привилегия «узнать среднюю температуру по больнице» — всему персоналу). Таким образом, при появлении нового сотрудника или перемене служебного положения старого достаточно включить его в одну из зажанных ролей, не меняя соответствия ролей и привилегий.

Для большего облегчения администрирования и более естественной структуры вводится *иерархия ролей*, а именно, для каждой роли указываются роли ей подчиненные (в примере с больницей роль медсестры подчинена роли хирурга, а роль хирурга — роли главврача). Каждой ро-

ли может быть подчинено несколько ролей и каждая роль может быть подчинена нескольким ролям. Каждый пользователь обладает не только привилегиями его собственных ролей, но и привилегиями всех ролей ему подчиненных и более низких ролей. Таким образом, для каждой роли необходимо указывать лишь те привилегии, которые отличают ее от других ролей. В случае с больницей, например, роли лечащего врача необходимо дать только привилегии, связанные с больным, которого он обследует, но не имеет смысла давать привилегии, которые есть у медсестры, так как медсестра является подчиненной ролью.

Дадим теперь строгие определения введенным выше понятиям.

Определение 2. Ролевой моделью называется:

- U, множество пользователей;
- R, множество ролей и AR, множество административных ролей (необходимо, чтобы $R \cap AR = \varnothing$);
- P, множество привилегий и AP, множество административных привилегий (необходимо, чтобы $P \cap AP = \varnothing$);
- $PA \subseteq P \times R$, отношение ролей и привилегий и $APA \subseteq AP \times AR$, отношение административных ролей к административным привилегиям;
- $UA \subseteq U \times R \cup AR$, отношение пользователей и ролей и $AUA \subseteq U \times AR$, отношение пользователей и административных ролей;
- $RH \subseteq R \times R$, частичный порядок ролей и $ARH \subseteq AR \times AR$, частичный порядок административных ролей, в дальнейшем обозначаемые $\succeq$.

Тогда привилегия p дается пользователю u, если $\exists\, r, r' \in R$: $(u, r') \in UA$, $r' \succeq r$, $(p, r) \in PA$, то есть эта привилегия дается некой роли (r), подчиненной одной из ролей пользователя (r). Административные привилегии даются аналогично.

В определении употребляется понятие административной привилегии и роли, которые не упоминались ранее. Эти понятия появились после создания первой ролевой модели и, именно благодаря им, стала возможным реализация дискреционных политик в ролевой модели [1]. Административная привилегия — это возможность изменить отношения пользователи — роли и роли — привилегии. Формально данный факт можно определить следующим образом: с каждой административной привилегией ap связываются подмножества $UA_{ap} \subseteq U \times R \cup AR$, $PA_{ap} \subseteq P \times R$ и $APA_{ap} \subseteq AP \times AR$. Тогда пользователь, обладающий првилегий ap может дать или отнять у пользователя u роль r, если $(u, r) \in UA_{ap}$, аналогично он может дать или отнять у роли r привилегию p, если $(p, r) \in PA_{ap}$. Таким образом, каждой административной привилегии соответствует возможность изменить лишь часть отношений, определяющих политику безопасности. Введение административных ролей позволяет сильно упростить управление системой безопасности, так как без этих ролей и привилегий производить настройку мог лишь администратор. Например, ранее начальник отдела при перемещения своего сотрудника с должности должен был согласовывать с администратором всей компании назначение ролей этому сотруднику. В рамках новой модели он может решить этот вопрос самостоятельно.

3. Эквивалентность ролевой и дискреционной моделей

Одним из важных вопросов, встающих при объединении систем или переходе на новое программное обеспечение, является теоретическая возможность переноса формально-логического представления моделей контроля доступа. Положительный ответ на него означает, что существуют формально-логическое представление, позволяющее описать обе модели разных политик безопасности. Таким образом, вопрос переноса моделей разграничения доступа сводится к переносу из одного представления в другое, которое позволит в определенном выше смысле объединить политики обеих моделей.

3.1. Построение ролевой политики в дискреционной модели

В данном разделе мы будем рассматривать только политики без административных прав.

Для начала дадим точное определение эквивалентности для данного случая. Пусть нам дана ролевая политика, дискреционная политика и заданы отображения $U_S\colon U \to S$ множества пользователей ролевой политики в множество субъектов дискреционной политики и $P_A\colon P \to O \times A$ множества привилегий на множество доступов к объектам. При этом отображение U_S должно быть инъективным. Эти политики с введенной парой отображений будем считать эквивалентными, если пользователь $u \in U$ имеет привилегию p тогда, когда $U_S(u)$ имеет доступ $P_A(p)$ и, наоборот, субъект $s \in U_S(U)$ имеет доступ (o, a) тогда, когда пользователь в ролевой политике $U_S^{-1}(s)$ имеет привилегию (или привилегии) $P_A^{-1}(o, a)$.

Перейдем к построению политики. Зададим $O = \{o\}$, $S = U \cup \{u_0\}$ (где u_0 — некоторый новый пользователь, не встречающийся в U), а $A = P$. Матрицу M построим так: $(u, o, p) \in M \iff \exists\, r, r' \preceq r : (u, r) \in UA$, $(p, r') \in PA$. Кроме этого, назначим u_0 владельцем этого объекта. Отображения U_S и P_A зададим так: $U_S(u) = u$, $P_A(p) = (o,\ p)$.

Докажем эквивалентность по данному выше определению. Пусть пользователь u имеет привилегию p. Тогда из определения ролевой политики следует, что $\exists\ r,\ r' \preceq r : (u,\ r) \in UA,\ (p,\ r') \in PA$. Однако, тогда по построению матрицы доступа у субъекта $U_S(u) = u$ будет доступ p к объекту o (напомним, это единственный объект в нашей построенной политике), что и требовалось (так как $P_A(p) = (o,\ p)$).

Наоборот, если у субъекта u $(u \neq u_0)$ есть доступ p к объекту o, тогда $\exists\ r,\ r' \preceq r : (u,\ r) \in UA,\ (p,\ r') \in PA$, следовательно из определения ролевой политики у пользователя u есть привилегия p.

3.2. Построение дискреционной политики в ролевой модели

Теперь докажем более интересный обратный факт (имеющий более важное практическое значение): по заданной дискреционной политику можно построить ролевую эквивалентную ей. В данном случае нам придется расширить понятие эквивалентности, так как пользователи теперь имеют возможность модифицировать политику, используя административные привилегии.

Как и в предыдущем случае, нам заданы инъективное отображение $S_U\colon S \to U$ и отображение $P_A\colon P \to O \times A$. Если субъект s мог получить доступ a к объекту o, то пользователь $S_U(s)$ должен иметь привилегию (или привилегии) $P_A^{-1}(o,\ a)$ и наоборот, если пользователь $u \in S_U(S)$ имеет привилегию p, то субъект $S_U^{-1}(u)$ должен иметь доступ $P_A(p)$ Кроме того, если субъект s имеет право добавить или убрать доступ a к объекту o для другого субъекта s', то пользователь $S_U(s)$ должен иметь возможность так изменить отношение UA или AUA, чтобы пользователь $S_U(s')$ получил или перестал иметь привилегию $P_A^{-1}(o,\ a)$ и наоборот.

Построим такую политику и такие отображения. Множество пользователей зададим U равным множеству субъектов S и, соответственно, функцию S_U — тождественной. Для каждого объекта o и для каждого $a_i \in A \setminus \{c,\ ct\}$ зададим роль $r_{o,\ a_i}$ и две административные роли own_o и own_o^t. Соответственно, для каждого $a_i \in A \setminus \{c,\ ct\}$ задаются привилегии:

- p_{o,a_i}, возможность доступа a_i к o;

и административные привилегии:

- $\mathrm{grant}_{o,\ a_i}$, возможность добавить или убрать любого пользователя из роли $r_{o,\ a_i}$;
- $\mathrm{grant}_{o,\ c}$, возможность добавить или убрать любого пользователя из ролей own_o или own_o^t.

Соответственно, роли $r_{o,\ a_i}$ дается привилегия $p_{o,\ a_i}$, роли own_o даются привилегии $\mathrm{grant}_{o,\ a_i}$ для всех a_i, а роли own_o^t — $\mathrm{grant}_{o,\ c}$.

Множества ролей и привилегий являются объединениями построенных множеств по всем объектам. Роли упорядочены следующим образом: $\forall\ o\ \mathrm{own}_o^t \succeq \mathrm{own}_o$, все остальные пары несравнимы.

Функция P_A задается следующим образом: если $p = p_{o,\ a}$, то $P_A(p) = (o,\ a)$, если $p = \mathrm{grant}_{o,\ a}$, то $P_A(p) = (o,\ c)$, а если $p = \mathrm{grant}_{o,\ c}$, то $P_A(p) = (o,\ ct)$.

Назначим роли следующим образом: если у субъекта был доступ a к объекту o, то включим его в роль $r_{o,\ a_i}$, если у него был доступ c, то включим в роль own_o, а если ct, то в роль own_o^t.

Аналогично предыдущему пункту проверяется эквивалентность построенной политики и исходной.

Заключение

Итак, мы указали алгоритм построения эквивалентной ролевой политики для любой дискреционной и эквивалентной дискреционной для ролевой без административных привилегий. Полученный результат указывает на то, что, обладая всеми уже перечисленными ранее достоинствами, ролевая модель предоставляет не менее (а даже более) точный контроль доступа. Кроме отмеченного обстоятельства, переход систем с дискреционной политики на ролевую, можно сделать автоматизированным, что, несомненно, должно вызвать интерес администраторов, обсуждающих вопрос о переводе систем на новое программное обеспечение, строящееся на ролевой модели безопасности.

Список литературы

[1] *Ravi S. Sandhu*, *Edward J. Coyne*, *Hal L. Feinstein*, *Charles E. Youman*. Role-Based Access Control Models. IEEE Computer, 29, 2, 38–47.

[2] *David Ferrariolo*, *Richard Kuhn*. Role-Based Access Control. Proceedings of 15th National Computer Security Conference, 1992, 554–563.

[3] Department of Defense Trusted Computer System Evaluation Criteria. DoD, DoD 5200.28-STD, 1985.

[4] A Guide To Understanding Discretionary Access Control In Trusted Systems. National Computer Security Center, 1987.

[5] RBAC in the Solaris Operating Environment. Sun Microsystems, 2001

[6] eXtensible Access Control Markup Language (XACML) Commitee Specification. OASIS Open, 2003.

Анализ тенденций развития аппаратно-программного обеспечения и их влияния на информационную безопасность

В. Б. Бетелин, С. Г. Бобков, В. А. Галатенко, А. Н. Годунов, А. И. Грюнталь, А. Г. Кушниренко

В настоящей работе конкретизируются сформулированные в докладе [1] идеи борьбы со сложностью современного аппаратно-программного обеспечения как необходимого условия повышения информационной безопасности систем. Анализируются тенденции развития аппаратно-программного обеспечения и их влияние на информационную безопасность.

Введение

Как указано в докладе [1], современное аппаратное и программное обеспечение характеризуется, с одной стороны, возрастанием сложности (миллионы электронных компонентов, десятки миллионов строк исходного текста), а, с другой стороны, — сокращением сроков разработки, стремлением производителей к скорейшему выходу на рынок с новыми изделиями. Совокупность этих двух тенденций, очевидно, отрицательно сказывается на информационной безопасности современных систем, приводит к квадратичному нарастанию числа известных уязвимостей в наиболее широко используемых операционных системах (об этом свидетельствуют материалы [2]).

В такой ситуации попытки повысить информационную безопасность путем реализации в операционных системах дополнительных механизмов безопасности (например, мандатного управления доступом, с которыми можно ознакомиться в [3]) дают скорее противоположный результат, поскольку увеличивают сложность систем. Системы небезопасны не потому, что в них не хватает каких-то защитных средств; они уязвимы, потому что сложны. Требуются принципиально новые концептуальные и архитектурные решения.

1. Основные тенденции

С проблемой сложности специалисты по технологии программирования столкнулись в шестидесятых годах двадцатого века. Для решения это проблемы сначала было предложено структурированное программирование, затем, как его развитие, — объектно-ориентированный подход. Последний остается наиболее эффективным инструментом в технологии программирования больших систем и в настоящее время. К сожалению, в информационной безопасности он еще не стал общепринятым.

Для нас существенны следующие свойства объектов:

- инкапсуляция: наличие границ между объектами, невозможность манипулирования объектами в терминах их внутренней реализации;
- наличие у объектов определенных интерфейсов и построенных на их основе протоколов, специфицирующих правила обращения к объектам;
- потенциальная активность объектов, возможность их параллельной работы.

Объектно-ориентированные системы строятся в многоуровневой архитектуре, с небольшим числом сущностей на каждом уровне и с умеренным числом связей между объектами, что делает подобные системы познаваемыми и, следовательно, принципиально контролируемыми. Если учесть перечисленные выше свойства объектов, можно считать, что объектно-ориентированные системы имеют сетевую организацию, поддающуюся управлению.

В работе [4] предложена постановка задачи разграничения доступа в распределенной объектной среде. Когда объект-клиент выбирает объект-сервер для выполнения своих запросов, он задает ряд ограничений, которым должен удовлетворять вызываемый объект. Последний, в свою очередь, задает ограничения на вызывающий объект и, кроме того, налагает на себя так называемые добровольные ограничения. (Например, текстовый редактор может изменять только редактируемый файл.) Процесс выбора объекта-сервера аналогичен подбору подходящей схемы программы при автоматической генерации программ в рамках инструментальной системы ЭСКОРТ [5].

Использование методов технологии программирования при решении проблем информационной безопасности представляется весьма перспективным, но в настоящее время эти две области неоправданно разобщены.

Тенденции развития аппаратного обеспечения проанализированы до-

вольно давно и уже длительное время остаются неизменными. Каждые 18 месяцев доступная вычислительная мощность удваивается. Каждые 12 месяцев удваивается доступный объем дискового пространства. Таким образом, доступная вычислительная мощность возрастает на три десятичных порядка примерно за пятнадцать лет, а емкость накопителей — за десять лет. Это означает, что на аппаратуре можно не экономить, развертывая ее в необходимых количествах и проводя наращивание по мере необходимости.

2. Следствия для информационной безопасности

В докладе [6] предложен принцип разнесения доменов выполнения по узлам сети. Его можно сформулировать как «один домен выполнения на узел сети с одним сервисом.» Иными словами, на одном узле сети функционирует один информационный или защитный сервис, локальный доступ к узлам невозможен, сетевой производится по определенным, контролируемым протоколам, а коммуникации между узлами защищаются криптографическими средствами, на которые, помимо шифрования, возлагаются функции аутентификации и управления доступом.

На наш взгляд, сетевая организация систем является ключевым элементом обеспечения информационной безопасности. Можно сказать, что проблемы безопасности следует «выдавить» в сеть. Когда на одном узле сети функционирует несколько процессов, взаимодействие между которыми не подчиняется каким-либо протоколам, задача разграничения доступа к разделяемым ресурсам становится неразрешимой. Многолетний опыт и современное состояние исследований и практических разработок в области информационной безопасности демонстрируют это с достаточной очевидностью. Укажем хотя бы на проблему скрытых каналов и необходимость введения доверенных субъектов в модели Белла — Ла Падула [7], считающейся эталоном доказуемой безопасности.

Исторически локальная обработка данных появилась раньше распределенной. То же произошло и со средствами безопасности. Представляется, что в настоящее время соотношение между локальными и сетевыми средствами безопасности прямо противоположно тому, которое необходимо для современных распределенных систем. Локальные пользователи, которые, как правило, с практической точки зрения являются доверенными, подвергаются двойному контролю — операционной системы и используемых информационных сервисов. Операционная система обеспечивает несколько защитных рубежей — идентификацию/аутентификацию, разграничение доступа, протоколирование и аудит. Удаленные пользователи, в массе своей недоверенные, контролируются только приложениями, поскольку в операционных системах отсутствуют необходимые понятия (удаленные субъекты не считаются пользователями в смысле ОС со всеми вытекающими отсюда последствиями) и механизмы безопасности. Иными словами, локальная защита от доверенных пользователей является эшелонированной, а сетевая (от недоверенных) — нет. Если приложение, функционирующее на узле сети, содержит уязвимость (а ввиду сложности современных приложений это более чем вероятно), то в результате ее эксплуатации удаленный пользователь может вовлечь приложение в выполнение злоумышленных действий и получить полный контроль над узлом.

В этой связи трудно согласиться с мнением автора работы [8], считающего, что теоретические основы информационной безопасности давно сформировались, они отвечают практическим потребностям, и проблема лишь в том, что современные специалисты не владеют ими в достаточной степени. Фундамент объектно-ориентированной или, что то же, «сетецентрической» безопасности еще предстоит заложить.

Сетевая организация информационных систем может проявляться и использоваться различными способами. В общем виде ее следует трактовать как одно из приложений объектно-ориентированного подхода. Информационная система строится из объектов с необходимой степенью гранулярности, а границы между объектами по возможности делаются физическими, поддерживаемыми аппаратно (объекты разносятся по узлам сети). Таким образом, в качестве концептуальной основы обеспечения информационной безопасности систем предлагается их объектная организация с физическими границами между объектами.

Отмеченный выше рост доступной вычислительной мощности может быть реализован, в частности, путем построения систем с большим (порядка десятков и сотен тысяч) числом простых процессоров, связанных в сеть (аналогичные идеи изложены в [9]). Заметим, что простота процессора — необходимое условие его безопасности, отсутствия в нем уязвимостей. Это — крайняя степень гранулярности сетевой архитектуры. Предполагается, что на каждом процессоре функционирует один поток управления, у них нет общей памяти, все межпотоковые взаимодействия носят сетевой характер. Для защиты каждого процессора может быть выделена микросхема, обеспечивающая сетевую связность, выполняющая функции межсетевого экрана и криптографические операции. Тем самым полезная функциональность отделяется от защитной, что, в частности, позволяет решить проблемы производительности этих существенно разных компонентов.

У любой сети должен быть центр управления. Для больших конфигураций подсистема управления строится в многоуровневой архитектуре. Не должна быть исключением и сеть микропроцессоров. Центр управления обеспечивает загрузку программ на процессоры, задание правил фильтрации потоков данных и распределение криптографических ключей. Таким способом проводится в жизнь выбранная политика безопасности.

Если на процессоре выполняется один поток управления, то и сам процессор, и операционная система могут стать проще по сравнению с традиционными, современными. В частности, ОС может оставить себе только исторически первую функцию реализации виртуальной машины, более удобной для приложений, чем физическая.

Между двумя крайними точками спектра сетевых конфигураций. монолитными системами и сетью микропроцессоров — имеются промежуточные точки, которые можно рассматривать как этапы перехода на чисто сетевую архитектуру. Целесообразным представляется организация одной вы числительной системы в виде сети взаимодействующих, но не доверяющих друг другу (и пользователю) модулей — центрального процессора и оперативной памяти, сетевого, дискового, графического и других контроллеров (см. также [10]).

Монолитные системы сложны, заведомо содержат уязвимости и компрометируются целиком. Сетевые могут быть организованы проще, содержать меньше уязвимостей, дольше противостоять атакам злоумышленников, принимать дополнительные защитные меры, извещать пользователей и администраторов безопасности о компрометации отдельных узлов. Реализация интеллектуальных самозащищающихся модулей представляется весьма перспективным направлением практической информационной безопасности.

Объектная организация с физическими границами между объектами может применяться не только к аппаратным компонентам, но и к потокам данных (об этом можно посмотреть также [11]). Целесообразно физически разграничить пользовательские и административные потоки данных, потоки, связанные с деятельностью администраторов безопасности, внутренние и внешние потоки, потоки с разной политикой безопасности и т. п. У интеллектуальной самозащищающейся сетевой карты может быть по крайней мере три сетевых интерфейса. Через один из них проходят пользовательские данные, через другой осуществляется конфигурирование и чтение регистрационных данных, через третий может направляться сигнал тревоги администратору безопасности или осуществляться вмешательство последнего в работу системы.

Отметим, что организация описанной «многожильной» сети не обязательно требует прокладки множества дорогих кабелей. Стандарт безопасности для беспроводных сетей с инфраструктурой IEEE Std 802.11i [12] предусматривает достаточные защитные средства канального уровня.

Тенденция ежегодного удвоения емкости накопителей позволяет реализовать предложенный в рамках проекта ЭСКОРТ [5] принцип неуничтожения информации. При изменении объекта (такого, например, как дисковый файл или запись базы данных) создается его новая версия, а старая остается неизменной. Неуничтожение информации может быть обеспечено аппаратно, как одно из свойств интеллектуального самозащищающегося дискового контроллера. В результате станет принципиально невозможным уничтожение регистрационных данных, будет повышена устойчивость систем к ошибкам и отказам.

Заключение

Объектная организация с физическими границами между объектами — универсальный архитектурный принцип повышения информационной безопасности. Он является радикальным, действенным средством борьбы со сложностью информационных систем — основным источником уязвимостей. Анализ тенденций развития аппаратного и программного обеспечения подтверждает возможность поэтапной реализации этого принципа с получением практических результатов в ближайшем будущем.

Список литературы

[1] *Бетелин В. Б.* Проблемы безопасного программного обеспечения. Материалы конференции «Математика и безопасность информационных технологий» (МаБИТ-03). М.: МЦНМО, 2004, 12 с.

[2] *Lee S. C.*, *Davis L. B.* Learning from Experience: Operating System Vulnerability Trends. IEEE IT Pro, January/February 2003, 8 pp.

[3] Операционная система Linux с дополнительными средствами безопасности. `http://www.nsa.gov/selinux/`.

[4] *Галатенко А. В.*, *Галатенко В. А.* О постановке задачи разграничения доступа в распределенной объектной среде. Вопросы кибернетики. Информационная безопасность. Операционные системы реального времени. Базы данных. М.: НСК РАН, 1999, 11 с.

[5] *Бетелин В. Б.*, *Галатенко В. А.* ЭСКОРТ — инструментальная среда программирования. Юбилейный сборник трудов институтов Отделения информатики РАН. Том. II. Москва, 1993, 21 с.

[6] *Бетелин В. Б.*, *Галатенко В. А.*, *Годунов А. Н.*, *Грюнталь А. И.* Обеспечение информационной безопасности систем на программной плат-

форме ос2000. Материалы конференции «Математика и безопасность информационных технологий» (МаБИТ-03). М.: МЦНМО, 2004, 13 с.

[7] *Bell D. E.*, *La Padula L. J.* Secure Computer Systems: Mathematical Foundations and Model. Technical Report M74–244. The MITRE Corporation, 1973.

[8] *Schell R. R.* Information Security: Science, Pseudoscience, and Flying Pigs. Proceedings of the 17th Annual Computer Security Applications Cinference (ACSAC 2001). IEEE, 2001, 12 pp.

[9] *Weissman C.* MLS-PCA: A High Assurance Security Architecture for Future Avionics. Proceedings of the 19th Annual Computer Security Applications Cinference (ACSAC 2003). IEEE, 2003, 11 pp.

[10] *Ganger G. R.*, *Nagle D. F.* Better Security via Smarter Devices. Proceedings of the Eighth Workshop on Hot Topics in Operating Systems (HotOS-VIII'01). IEEE, 2001, 6 pp.

[11] *Bryant E.*, *Early J.*, *Gopalakrishna R.*, *Roth G.*, *Spafford E. H.*, *Watson K.*, *Williams P.*, *Yost S.* Poly Paradigm: A Secure Network Service Architecture. Proceedings of the 19th Annual Computer Security Applications Conference (ACSAC 2003). IEEE, 2003, 10 pp.

[12] Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications: Medium Access Control (MAC) Security Enhancement. IEEE, 2004.

Архитектура распределенных сетевых процессоров: особенности применения в системах информационной безопасности

В. С. Заборовский, Ю. А. Шеманин

Введение

Современные телематические сети представляют собой распределенные иерархические структуры, состоящие из узлов и линий связи. Узлами сети являются отельные компьютеры с сетевыми интерфейсами, используемыми для обмена данными. Узел с несколькими сетевыми интерфейсами называется маршрутизатором или сетевым процессором (СП). Каждый сетевой интерфейс имеет один или несколько идентификаторов, которые называются адресами. Множество адресов образует наделенное метрикой сетевое пространство. Топология этого пространства определяется линиями связи, которые связывают не менее двух сетевых адресов. Если число адресов больше, чем два, то линия связи называется широковещательной, если адреса, которые связывает линия, совпадают, то линия образует замкнутую петлю. Расстояние между узлами определяется числом линий связи между ними. Целью объединения узлов в сеть является организация обмена данными между сетевыми приложениями, которые выполняются в узлах сети. Обмен данными осуществляется путем посылки и приема сетевых пакетов. Пакет, это специальная логическая последовательно-рекурсивная структура, которая формируется в узлах сети для организации обмена данными. Последовательная часть этой структуры состоит из двух полей — заголовка и данных. Рекурсивность пакета связана с тем, что данные могут быть другим пакетом со своей определенной структурой.

Правила обмена данными в телематической сети состоят в том, что в процессе передачи данных выбор маршрута осуществляется в узлах сети на основе информации, содержащихся в заголовках этих пакетов. В самой линии связи никакой обработки пакета не происходит. Результатом обработки является решение о выборе сетевого интерфейса

маршрутизатора, с которого данный пакет будет отправлен в сеть. Если в процессе обработки принято решение не отправлять пакет в сеть, то считается, что пакет достиг заданного сетевого узла, либо будет утерян. Таким образом, базовая функциональность сетевого маршрутизатора определяется двумя последовательными стадиями обработки пакетов после их приема из линии связи, а именно «запомнить — отправить». В работе рассматривается новый подход к выбору архитектуры СП, когда расширение функциональных требований к различным стадиям обработки пакетов, в частности, при решении задач информационной безопасности, реализуется путем распределения процедур их реализации между отдельными устройствами. Особенностью предлагаемого разделения является то, что оно не нарушает существующие адресные связи между узлами сети. Адресная инвариантность преобразований узлов, при расширении их функциональности достигается путем использования специального режима функционирования сетевых узлов, получившего название «стелс».

1. Тенденции развития систем телематики

По мере роста скорости передачи информации по коммутационным линиям и расширения спектра протоколов возрастают требования к производительности процессоров, используемых в узлах сети для обработки пакетов. Архитектура и особенности функционирования таких устройств обработки или сетевых процессоров (СП) стала предметом большого числа исследований и разработок [1]–[3]. В своем большинстве этих исследований не были основаны на систематическом изучении специфических требований и проектных альтернатив, а использовали известные результаты применения многопроцессорных архитектур для повышения производительности обработки потоковых данных. Предложенные решения по повышению функциональности узлов маршрутизации, основывались на использования параллельных и конвейерных процедур для организации трех стадий обработки пакетов по формуле «запомнить — обработать — отправить».

В общем случае все решения можно разделить на два класса. Во-первых, это решения направленные на повышение производительности работы устройств маршрутизации. Основными параметрами, управляющими их работой, являются адреса приемника пакетов, поэтому применяемые решения направлены на ускорение поиска данных в таблицах маршрутизации. Второй класс решений связан с реализацией различных процедур классификации пакетов, обработки данных, обеспечения качества обслуживания, распределением полосы пропускания

и пр. В принципе такое разделение процессов обработки позволяет перераспределить интегральную производительность сетевого узла на отдельные составляющие не только на уровне отдельных процессов, но и на уровне выделенных аппаратно-программных компонент. Если для обмена данными между такими компонентами не используются сетевые адреса, то функционально их можно отнести к каналу связи, что приводит к новой формуле функционирования сети «обработать — запомнить — отправить».

2. Телематические системы информационной безопасности

В основе современной концепции компьютерных телекоммуникаций лежит принцип пакетной коммутации. На практике применение этого принципа сводится к использованию модели взаимодействия открытых систем (МВОС), в которой выделяется несколько уровней управления. На каждом из этих уровней объектом управления являются специальные структуры данных, которые называются пакетами. Процесс генерации пакетов сводится к конвейерной процедуре. В этой процедуре можно выделить следующие стадии:

1) выделение определенного объема данных, подлежащих передаче через сеть;
2) формирование структуры, в которой производится количественный учет всего объема передаваемых данных;
3) присоединение к данным специального заголовка, содержащего набор параметров, на основании которых производится обработка пакета в узлах сети;
4) формирование кадра, структура которого соответствует требованиям каналообразующей аппаратуры;
5) передача кадра по коммутационной линии, которая связывает два узла сети.

В процессе передачи пакетов различают несколько типов сетевых узлов, а именно: узлы генерации, узлы в которых производится обработка только заголовков пакетов; узлы, в которых производится обработка заголовков и данных. Процесс маршрутизации или выбор интерфейса, куда передается пакет после обработки, носит локальный характер, то есть осуществляется в каждом узле сети, через который проходит пакет. Для маршрутизации используется адрес узла получателя пакета, который указан в соответствующем поле заголовка и таблица связи между адресами узлов сети и номерами интерфейсов маршрутизатора.

Описанный выше процесс весьма уязвим для различного рода воздействий, которые могут нарушить стандартную процедуру передачу пакетов или произвести подмену пакетов на этапе их следования от узла генерации до узла приема.

Основными мерами защиты являются:

1) организация специальной траектории движения пакетов через такие узлы сети, в которых реализуются специальные правила обработки, позволяющие не допустить прохождение через них пакетов с определенными адресами и параметрами заголовков;
2) использование режима туннеля, когда защищаемый пакет передается в поле данных другого сетевого пакета;
3) использование специальных режимов передачи пакетов, когда параметры заголовков защищены криптографическими средствами.

3. Сетевые процессоры

В настоящее время индустрия телекоммуникаций, обладая избыточной пропускной способностью физических каналов связи, испытывает постоянно возрастающую потребность в эффективных средствах обработки пакетов для маршрутизации и защиты информации. Эти потребности стимулировали широкий спектр исследований по разработке специальных вычислительных устройств, используемых в процессе обработки пакетов. При разработке современных СП должны учитываться тенденции роста пропускных способностей линий связи при использовании оптических сред и применении технологий волнового мультиплексирования. Общие решения задачи повышения производительности вычислителей можно разделить на следующие группы: создание СП на базе параллельных процессоров, использующих разделяемую оперативную память; разработка конвейерных СП, с ресурсами оперативной памяти, распределенными между отельными фазами обработки; гибридные конвейерно-параллельные архитектуры, в которых стадии последовательной и параллельной обработки согласуются с количеством независимых потоков данных.

Эффективность таких решений целиком определяются алгоритмическими особенностями решаемых задач и способом доставки для них данных. В случае обработки сетевых пакетов существенное значение имеют следующие обстоятельства: потоковый характер данных, при котором число одновременно обрабатываемых соединений зависит от числа узлов с различными сетевыми адресами; последовательный способом передачи пакетов в потоке. Так как передача пакетов осуществляется в

асинхронном режиме, то есть инициируется каждым узлом независимо, то количество проходящих через маршрутизаторы логических соединений является случайной величиной с фрактальной функцией распределения [2]. Сложный характер процессов пакетной коммутации приводит к тому, что номинальное число параллельных процессоров в архитектуре СП не определяет полностью его производительность, а оптимальное количество стадий конвейерной обработки зависит от характера решаемой задачи и может меняться. Все эти обстоятельства являются предпосылкой для разработки новых подходов к организации процессов обработки сетевых пакетов.

4. Применение распределенных СП системах защиты

Разработка СП для систем защиты может быть основана на разделении функций обработки пакетов по принципу базовых и дополнительных операций. К базовым операциям относится маршрутизация пакетов, а к дополнительным — остальные операции с пакетами, связанные с расширением функциональности СП, например фильтрация пакетов. Предлагаемое разделение позволяет рассматривать сетевой узел как часть специальной сети обработки пакетов. Топология соединения устройств обработки должна быть такой, чтобы в процессе передачи между ними пакетов не использовались адреса узлов, входящих в таблицу маршрутизации.

Применительно к задачам информационной безопасности применение данного подхода позволяет использовать технологию сетевого управления, основанную на использовании системного принципа «безопасность через защиту средств защиты». Этот принцип уравнивает значимость двух ключевых аспектов информационной безопасности, положенных в основу ГОСТ 15408, а именно функциональность и доверие. Следование этому принципу означает, что средства, используемые для защиты информации в компьютерных сетях, должны обладать эффективными механизмами обеспечения собственной безопасности, как на стадии разработки (контроль за отсутствием недекларируемых возможностей — НДВ), так и на стадии оперативного функционирования. Для этого на нескольких уровнях модели взаимодействия открытых систем (ВОС) должны быть обеспечены меры, обеспечивающие невозможность локализации места размещения в сети устройств защиты методами удаленного мониторинга. Такая скрытность функционирования приводит к изменению модели защиты, так как большинство существующих средств организации сетевых атак и разрушающих воз-

действий построена на удаленной нейтрализации устройств, используемых для защиты информационных ресурсов в сети.

Создать средства защиты на базе распределенных СП с использованием режима «стелс» возможно потому, что устройства защиты в большинстве режимов функционирования не являются источниками или конечными получателями сетевых пакетов. Поэтому сетевые интерфейсы этих устройств могут не иметь физических и логических адресов, следовательно, характер прохождения через них IP пакетов или MAC кадров аналогичен прохождению их через сетевые концентраторы (HUB) и сегменты кабельных линий, используемых в процессе межсетевого обмена. Применение данного метода сокрытия сетевого адреса расположения средств информационной безопасности с одной стороны обеспечивает выполнение функций защиты, а с другой из-за отсутствия адресов у сетевых интерфейсов устройств обработки пакетов не требует изменения топологии сетевых связей и ранее принятой политики маршрутизации пакетов. Устройства защиты, использующие технологию «стелс» обладают рядом преимуществ не только с точки зрения скрытного характера своего функционирования, но также в аспекте масштабирования производительности и повышения уровня надежности работы. Повышение производительности основано на использовании последовательно-параллельного характера сетевого трафика, когда независимые логические соединения формируются путем последовательной передачи отдельных пакетов с определенными адресами источников и приемников сообщений. В результате появляется возможность уменьшения задержек пакетов в цепочке операций «фильтрация, шифрование, передача пакета в сеть» и «прием пакетов из сети, фильтрация и дешифрование» за счет объединения сетевых процессоров в специализированный вычислительный кластер. Учитывая, что для типовой технологией передачи информации в настоящее время является широковещательная или коммутируемая рассылка кадров **IEEE 802.3** Ethernet, применение режима сокрытия адресов в сочетании с технологией параллельных вычислений позволяет организовать процесс обработки пакетов в ядре операционной системы используемых сетевых процессоров. Обработка пакетов на этом уровне без использования стека **TCP/IP** снижает уровень флуктуаций задержек из-за буферизации пакетов данных, способствуя дополнительному сокрытию места расположения средств защиты, что позволяет снизить риск несанкционированного использования информационных ресурсов за счет устранения множественного характера применения паролей доступа и функции авторизации пользователей. Многие из описанных выше особенностей применения распределенных СП реализованы в

серийно выпускаемом в НПО РТК межсетевом экране **ССПТ1М**, имеющем сертификат ГТК № 702.

Заключение

Создание сетевых процессоров с распределенной архитектурой позволят существенно расширить функциональность средств защиты информации. При этом дополнительные процедуры обработки могут быть интегрированы в стандартный процесс коммутации пакетов без изменения ранее принятой политики маршрутизации, что снижает затраты на модернизацию сети и способствует эффективному масштабированию используемых технических решений.

Список литературы

[1] *Vladimir Zaborovsky*. Multiscale Network Processes: Fractal and p-Adic Analysis. Proceedings of 10th International Conference on telecommunications (ICT'2003), University of Haute Alsace, Colmar, France, 2003.

[2] *Вильчевский Н. О.*, *Заборовский В. С.*, *Клавдиев В. Е.*, *Шеманин Ю. Е.* Методы оценки эффективности управления и защиты транспортных соединений в высокоскоростных компьютерных сетях. Материалы конференции «Математика и безопасность информационных технологий (МаБИТ-03)», МГУ им. М. В. Ломоносова, 23–24 октября 2003 г.

[3] *Vladimir Zaborovsky*, *Yuri Shemanin*, *Jim A. McCombs*, *Alex Sigalov*. Firewall Network Processors: Concept, Model and Platform. Proceedings of International Conference on Networking (ICN'04), Guadeloupe, 2004.

Проактивная безопасность вычислительных систем

О. В. Казарин

Введение

Рассматриваются методологические вопросы создания вычислительных систем, структурно несущих в себе защитные свойства. Лежащие в основе методы обеспечения безопасности вычислительных, функциональных и информационных ресурсов могут основываться на идеях из теории конфиденциальных вычислений, которые предусматривают построение надежных и безопасных вычислительных систем, даже если некоторые из участников проекта являются злоумышленниками.

При решении проблемы обеспечения безопасности вычислительной системы необходимо исходить из того, что наиболее вероятным информационным объектом воздействия будет выступать программное обеспечение, составляющее основу комплекса средств получения, семантической переработки, распределения и хранения данных, используемых при эксплуатации вычислительных систем.

В настоящее время одним из наиболее опасных средств информационного воздействия на компьютерные системы являются компьютерные вирусы, алгоритмические и программные закладки. Данные программные средства деструктивного воздействия по своей природе носят, как правило, разрушительный, вредоносный характер, а последствия их активизации и применения могут привести к значительному или даже непоправимому ущербу в тех областях человеческой деятельности, где применение компьютерных систем является жизненно необходимым. В связи с этим такие вредоносные программы будем называть разрушающими программными средствами. Особенности применения этих средств подробно исследуются в работах [1], [2].

Алгоритмические и программные закладки, вносимые на ранних этапах жизненного цикла вычислительной системы, имеют широкий спектр воздействий на программное обеспечение вычислительной системы и на информацию, обрабатываемую вычислительными средствами на этапе их эксплуатации. Следовательно, еще на этапах проектирования и создания вычислительных систем необходимо осуществлять мероприятия

по обеспечению их проактивной безопасности (в части создания программного обеспечения — это мероприятия по обеспечению его технологической безопасности [1]–[3]), когда в компоненты вычислительной системы при ее разработке вносятся защитные процедуры и механизмы.

1. Проактивная безопасность вычислительных систем

1.1. Общая постановка задачи

Основной отличительной особенностью предлагаемого подхода, связанного с созданием проактивно (априорно) безопасных вычислительных систем является то, что начало процесса обеспечения безопасности вычислительной системы при их разработке переносится на этапы, предшествующие этапу испытания программ. Тем самым, увеличивается общее время на внесение в вычислительную систему защитных функций. Иными словами, ключом к разработке проактивно безопасных вычислительных систем является стремление защищаться от разрушающих программных средств с самого начала жизненного цикла, а не после факта их создания.

Рассмотрим постановку задачи создания проактивно безопасных вычислительных систем на примере программного обеспечения, разрабатываемого в интересах таких систем. Предположим, что некоторому разработчику (коллективу разработчиков) предписывается разработать программу **П** для некоторой вычислительной системы. При этом последствия некорректного функционирования программы таковы, что могут привести к неким «нежелательным» или даже катастрофическим последствиям для объекта автоматизации, на котором функционирует данная вычислительная система. Исходя из гипотезы, что в общем случае проблема доказательства безопасности для сложных комплексов программ является неразрешимой [1], [2], неформальная качественная постановка задачи разработки защищенных программ для вычислительной системы может быть тогда сформулирована следующим образом.

Постановка 1. При некоторых условиях и ограничениях необходимо разработать программу **П**, которая корректно вычисляет результат *почти* для всех своих входных значений.

Таким образом, разработчик констатирует факт, что лишь для незначительной (возможно достаточно малой) доли входных значений программа может выдать некорректное выходное значение в результате необнаруженного программного дефекта или активизации необнаруженной программной закладки.

Постановка же задачи разработки проактивно безопасного программного обеспечения вычислительной системы меняется кардинальным образом.

Постановка 2. При некоторых условиях и ограничениях необходимо разработать программу $\mathbf{П}^*$, которая корректно вычисляет результат для всех своих входных значений с *пренебрежимо малой вероятностью ошибки*.

Одним из принципиальных условий решения задачи создания проактивно безопасной вычислительной системы в такой постановке является наличие набора определенных трансформаций процесса разработки, позволяющих переходить от традиционного пути создания вычислительной системы, которая будет затем проверяться на наличие разрушающих программных средств, к проактивно безопасным системам. К числу таких методов относятся методы конфиденциальных вычислений, методы создания самокорректирующихся и самотестирующихся программ и др. [1], [2].

1.2. Обобщенный портрет нарушителя

Рассматриваются два обобщенных типа нарушителей. Различия между ними заключается в их способности вмешиваться в процесс проектирования и создания вычислительной системы. Нарушитель, вмешивающийся в этот процесс (*активный нарушитель*), может:

- внедрять злоумышленников в коллективы, разрабатывающие различные компоненты вычислительной системы;
- внедрять злоумышленников, способных в совершенстве изучить «слабые» места вычислительной системы и особенности используемых технологий;
- внедрять неоптимальные информационные технологии;
- осуществлять злоумышленный выбор нерациональных алгоритмов работы;
- внедрять и использовать информационные технологии или их элементы, содержащие программные закладки;
- делать все, что может облегчить внесение закладок и затруднить их обнаружение;
- осуществлять поставку вычислительных средств, содержащих программные, аппаратные или программно-аппаратные закладки;
- формировать программные закладки в компоненте вычислительной системы, воздействующие на другие компоненты системы;
- организовывать маскирование спускового механизма программной закладки и др.

Нарушитель, наблюдающий за процессом проектирования и создания вычислительной системы, назовем его *пассивным нарушителем*, может:

- получать конфиденциальную информацию о характеристиках процесса разработки вычислительной системы;
- идентифицировать информацию с конкретными разработчиками и потенциальными пользователями;
- получать информацию о характеристиках трафика взаимодействия в распределенной вычислительной системе;
- считывать пароли, ключи и т. п. и отождествлять их с конкретными разработчиками и потенциальными пользователями и др.

Таким образом, нарушителя будем рассматривать как субъект (субъекты), совершающий несанкционированный доступ к информационному и функциональному ресурсу при проектировании и создании вычислительной системы с целью совершения самого широкого набора действий злоумышленного характера.

1.3. Общие подходы к созданию проактивно безопасных вычислительных систем

Ниже формулируются основные принципиальные свойства и требования к проактивно безопасным вычислительным системам, функционирование которых будет происходить при определенных ограничениях и допущениях.

1. *Свойство дистрибутивности компонентов вычислительной системы*. На этапах проектирования и создания проактивно безопасных вычислительных систем необходимо осуществлять распределение задач, ресурсов и ответственности между компонентами вычислительной системы.

2. *Свойство функциональной эквивалентности*. Проактивно безопасная вычислительная система должна быть функционально эквивалентна исходной вычислительной системе.

3. *Свойство оптимальности затрат*. Проактивно безопасная вычислительная система должна лишь незначительно снижать эффективность функционирования целевой вычислительной системы.

4. *Свойство устойчивости к сбоям*. Система должна оставаться полностью безопасной, даже если несколько ее компонентов становятся недееспособными с точки зрения обеспечения безопасности вычислительных, функциональных и информационных ресурсов системы.

5. *Свойство автоматического восстановления*. Система должна предусматривать возможность автоматического (возможно быстрого

автоматизированного) восстановления компонентов вычислительной системы и систематического применения механизмов автоматического восстановления.

6. *Свойство полного восстановления*. Как только скомпрометированный компонент системы восстановлен после сбоя, этот компонент будет опять «вносить свой вклад» в общую безопасность системы.

7. *Свойство конфиденциальности восстановления*. Система должна предусматривать возможность управления процессом восстановления компонента вычислительной системы посредством других (новых) конфиденциальных параметров (ключей, паролей, аутентификаторов и др.), которые не известны нарушителю. Это необходимо для того, что даже, если нарушитель теряет контроль над компонентом системы, он все еще может иметь достаточное количество данных о внутренних характеристиках этого компонента.

Приведем также ограничения и допущения при функционировании проактивно безопасных вычислительных систем:

- проактивная безопасность вычислительной системы сохраняется только в условиях, когда ее компоненты не подвергаются длительным деструктивным воздействиям;
- технические средства, на которых реализуется вычислительная система, не свободны от аппаратных закладок;
- количество компонентов вычислительной системы, подвергающихся деструктивным воздействиям, не должно превышать в некоторый момент времени предопределенно установленную границу.

Таким образом, создание проактивно безопасных систем с полным набором вышеперечисленных свойств (возможно лишь с их частью) с установленной совокупностью ограничений и допущений является центральным звеном процесса разработки современных вычислительных систем, применяемых на различных объектах информатизации. К потенциальным методам создания таких систем относятся научно-практические методы конфиденциальных вычислений.

2. Методология конфиденциальных вычислений

2.1. Конфиденциальные вычисления в сети процессоров

Теория конфиденциальных вычислений и ее основная составляющая — теория конфиденциального вычисления функция [4]–[8] являются фундаментальной основой для изучения многих научных и практических направлений в криптографии таких, как шифрование, аутентификация, схемы привязки, доказательства с нулевым разглашением и др. Кроме того, конфиденциальные вычисления являются мощным

средством для изучения широкого спектра задач распределенных вычислений.

Задачу конфиденциального вычисления функции, которая решается посредством многостороннего интерактивного протокола можно описать в следующей постановке. Имеется n участников протокола или n процессоров вычислительной системы, соединенных сетью связи. Изначально каждому процессору известна своя «часть» некоторого входного значения x. Требуется вычислить $f(x)$, f — некоторая известная всем участникам вычислимая функция, таким образом, чтобы выполнились следующие условия:

- *корректности*, когда значение $f(x)$ должно быть вычислено правильно, даже если некоторая ограниченная часть участников произвольным образом отклоняется от предписанных протоколом действий;
- *конфиденциальности*, когда в результате выполнения протокола не один из участников не получает никакой дополнительной информации о начальных значениях других участников (кроме той, которая содержится в вычисленном значении функции).

Можно представить следующий сценарий использования этой модели для разработки безопасного программного обеспечения. Имеется некоторый процесс, для управления которым необходимо вычислить функцию f. При этом последствия вычисления неправильного значения таковы, что представляется целесообразным пойти на дополнительные затраты, связанные с созданием сети из n процессоров и распределенного алгоритма для вычисления f. В системе имеется еще один абсолютно надежный участник, который имеет доступ к конфиденциальному значению x и имеет возможность выделить каждому участнику свою «долю» x. Название протоколы «конфиденциальное вычисление функции» отражает тот факт, что требование конфиденциальности является основным, т. е. значение x не должно попасть в руки злоумышленника.

2.2. Топология сети процессоров

Организация сети процессоров основана на предположении, что существует множество одинаковых или почти одинаковых процессоров, каждый из которых связан регулярной структурой с другими процессорами.

Например, процессоры могут быть вершинами полного двоичного дерева, и в этом случае процессор связывается с двумя дочерними и предшествующей вершиной, листья связаны только с предшествующими вершинами, корень, конечно, связан только с дочерними вершинами.

В другом примере, процессоры могут быть вершинами d-мерного гиперкуба. Гиперкубом размерности d является множество из $n = 2d$ вершин, связанных в форме d-мерного куба. Иначе говоря, пусть вершины представлены двоичными числами $a_1, a_2, \ldots, a_d$, где каждый разряд f_i равен 0 или 1. Пусть теперь вершины $a_1, a_2, \ldots, a_d$ и $b_1, b_2, \ldots, b_d$ связаны ребром тогда и только тогда, когда существует точно одно значение i, для которого $a_i \neq b_i$. Таким образом, когда $d = 2$, граф является квадратом, а когда $d = 3$, — это обычный куб.

В силу ряда причин, в том числе и связанных с решением задач защиты процесса вычислений от ошибочных и/или злонамеренных действий, в основном нас будет интересовать топология сети процессоров по схеме «каждый-с-каждым», по схеме с общей шиной, либо их объединение. Может также рассматриваться топология сети процессоров по схеме «с коммутатором» [1]. Следует также отметить, что можно использовать и другие топологии сети процессоров при условии, что каждый из них может иметь транзитный канал связи для других процессоров. Однако в этом случае взаимодействие процессоров при конфиденциальных вычислениях возможно только в асинхронных сценариях.

2.3. Обобщенные модели для сети синхронно и асинхронно взаимодействующих процессоров

Универсальная модель взаимодействия. Взаимодействие процессоров может осуществляться посредством конфиденциальных и/или открытых каналов связи. Каждые два процессора могут быть иметь симплексное, полудуплексное или дуплексное соединение. При этом каждое из соединений, в зависимости от решаемой задачи, в некоторый промежуток времени, может быть либо конфиденциальным, либо открытым.

Кроме того, может существовать широковещательный канал связи, то есть канал, посредством которого один из процессоров может одновременно распространять свое сообщение всем другим процессорам вычислительной системы. Такой канал еще называется каналом с общей шиной.

Взаимодействие в сети характеризуется трафиком, который может представлять собой совокупность сообщений, полученных участниками вычислений в некотором раунде протокола в установленной модели взаимодействия.

Обобщенная модель сбоев. Рассматривается сеть взаимодействующих процессоров. Некоторые процессоры могут «сбоить». При сбоях, приводящих к останову (FS-сбоях), сбоящий процессор может приостановить в некоторый момент времени отправку своих сообщений. В то же время предполагается, что сбоящие процессоры продолжают получение сообщений и могут выдавать «некую информацию» в свои выходные каналы. При Византийских сбоях (By-сбоях) процессоры могут произвольным образом «сотрудничать друг с другом» с целью получения необходимой для них информации или с целью нарушения процесса вычислений. При By-сбоях сбоящие процессоры могут объединять свои входы и изменять их. В то же время это должно происходить при условии невозможности изучения любой информации о входах несбоящих процессоров.

Сбои могут быть статическими и динамическими. При статических сбоях множество сбоящих процессоров фиксировано в начале и процессе вычислений, при динамических — множество сбоящих процессоров может меняться, как может меняться и характер самих сбоев. Сбоящие процессоры будут также называться нечестными участниками вычислений, в противоположность честным участникам, которые выполняют предписанные вычисления.

Обобщенная модель противника. Противника можно представлять как некий универсальный процессор, действия которого заключаются в «желании» получить значение функции, не зная большинство ее аргументов, или в «желании» получить значение функции по своему усмотрению (т. е., такого значения функции, которое его бы «устраивало»). В связи с этим, противник заинтересован:

- в получении конфиденциальной информации о входах несбоящих процессоров;
- в стремлении «сделать» процессоры сети сбоящими («подкупить» их).

Существует несколько обобщенных критериев, по которым можно оценивать противника в конфиденциальных вычислениях. К ним относятся:

- вычислительная мощность противника;
- контроль над коммуникациями;
- синхронизация;
- количество сбоящих процессоров;
- контроль над сбоящими процессорами;
- адаптивность при принятию решения о «подкупе» процессоров.

Таким образом, мы различаем противника, который имеет неограниченное время вычислений (экспоненциальный противник) и противника, который ограничен вероятностным полиномиальным временем (полиномиальный противник). При этом процессоры рассматриваются как

вычислительные устройства, работа которых также ограничена вероятностным полиномиальным временем.

Мы различаем три уровня контроля противника над коммуникациями. В большинстве случаев противник не имеет доступа к каналам связи, т. е., любые два несбоящих процессора могут взаимодействовать друг с другом без «боязни» прослушивания их противником и нарушения им связи между процессорами. В противном случае мы имеем дело с противником, который либо может только прослушивать сообщения, циркулируемые в сети (*прослушивающий противник*), либо с противником, который может знать все о внутренней конфигурации сети, может читать и изменять все сообщения сбоящих процессоров и может выбирать сообщения, которые будут посылать сбоящие процессоры при вычислениях (*вмешивающийся противник*).

В случае с прослушивающим противником мы выдвигаем предположение о наличии неконфиденциальных каналов, в случае с активным противником — предположение о наличии неаутентифицированных каналов. В любом другом случае мы выдвигаем предположение о наличии полностью защищенных каналов связи.

Рассматривается сеть взаимодействующих процессоров, либо снабженная общими глобальными часами (синхронизатором), либо сеть, таких часов не имеющая. В таком случае мы различаем синхронные конфиденциальные вычисления, либо асинхронные конфиденциальные вычисления. В асинхронных конфиденциальных вычислениях противник имеет дополнительную мощность по осуществлению злонамеренных действий.

По аналогии со сбоями противник может быть динамическим и статическим. *Статическим противником* является противник, который «сотрудничает» с фиксированным количеством сбоящих процессоров («подкупленных» противником). При действиях *динамического противника* количество сбоящих процессоров может меняться (в том числе непредсказуемым образом). Кроме того, противник может быть с априорными и апостериорными протокольными и раундовыми действиями, т. е., такой противник может выполнять злоумышленные действия не только в течение раунда (в его начале и конце) протокола, но и до начала выполнения протокола и после его завершения.

Если противник может в начале протокола «подкупить» несколько процессоров, то он может изучить информацию, получаемую ими в протоколе, и принять решение «подкупить» ли ему в следующий раз новый процессор, или нет в зависимости от той информации, которую он получил ранее в процессе вычислений. Такой динамический противник называется *адаптивным противником*. Следовательно, противник, который не имеет такой априорной информации перед принятием решения о «подкупе» процессоров, называется неадаптивным.

При действиях *мобильного противника* рассматриваются случаи, когда в определенный момент времени процессоры могут быть сбоящими, а в другой момент времени становятся снова несбоящими и не существует ограничений на общее количество процессоров, которые могут быть сбоящими в процессе вычислений.

Противник называется *t-противником*, если он «сотрудничает» с t процессорами.

Получестные модели поведения. Обычно основная цель при конфиденциальных вычислениях состоит в том, чтобы разработать протоколы, которые могут противостоять любому возможному поведению противника. Это делается, как правило, за два этапа. Сначала рассматривается так называемый «удобный» противник и строятся протоколы, которые являются безопасными в отношении такого противника. Затем, показывается, как преобразовать любой протокол, безопасный в получестной модели, в протокол, который является безопасным против любого возможного поведения противника.

В получестной модели имеется участник, именуемый получестным процессором, который следует предписанным протоколом действиям, за исключением того, что он может хранить информацию обо всех своих промежуточных вычислениях. Фактически достаточно хранить только внутренние случайно сгенерированные параметры (результаты подбрасывания монеты) и все сообщения, полученные от других процессоров. Таким образом, получестный процессор «подбрасывает честную монету» (генерирует случайный бит с вероятностью 1/2) и посылает сообщения в соответствии с инструкциями специальной программы (как функции от входа, результатов подбрасываний монеты и входных сообщений).

Различают три вида получестного поведения. В первом случае получестный процессор хранит все промежуточные внутренние данные (старые данные), полученные им в процессе вычислений. Таким образом построенный получестный процессор называется необнуляющимся (такое поведение наиболее трудно для обнаружения). Кроме того, получестный процессор, который в локальных (внутренних) вычислениях отклоняется от предопределенных инструкций, однако это не обнаруживается никаким дополнительным тестом, проводимым другими процессорами, называется получестным процессором, не обнаруживаемым тестами. Получестный процессор, который в локальных (внутренних) вычислениях отклоняется от предопределенных инструкций, однако это не обнаруживается никаким внешним тестом,

называется получестным процессором, не обнаруживаемым внешними тестами.

Злонамеренные модели поведения. Теперь будем рассматривать независимое отклонение процессоров от предопределенных протоколом действий. Для этого необходимо сделать несколько замечаний. Во-первых, для противника не существует никакого способа принудить несбоящие процессоры участвовать в выполнении каких-либо действий по его усмотрению при выполнении общего протокола. Во-вторых, необходимо отметить, что не существует никакого способа для противника зафиксировать по своему усмотрению вход протокола.

В злонамеренной модели сбоящие процессоры могут отклоняться указанным ниже образом от общего протокола:

- процессоры могут отказаться участвовать в протоколе, когда он инициируется;
- процессоры подменяют свои локальные входы (и могут, например, участвовать в протоколе с входами других процессоров);
- процессоры преждевременно прерывают протокол (например, после посылки своего последнего сообщения).

Теперь будем предполагать, что процессоры имеют в своем распоряжении доверенное третье лицо. Определим его как доверенный процессор или *TP*-процессор. Такая модель вычислений называется идеальной моделью. В идеальной модели сбоящему процессору позволяется отказаться участвовать в протоколе или подменять свои локальные входы. Более подробно идеальный и реальный сценарии для различных моделей вычислений рассматриваются далее. Рассматривается также реальная модель, в которой выполняется реальный протокол (и в ней не существует никаких доверенных третьих лиц). В этом случае, сбоящие процессоры могут следовать любой возможной стратегии.

При доказательстве безопасности в конфиденциальных вычислениях требуется эффективное преобразование противника для реальной модели в противника для идеальной модели.

3. Синхронная модель вычислений

3.1. Общее описание модели

Рассматривается сеть процессоров, функционирование которой синхронизируется общими часами (синхронизатором). Каждое локальное (внутреннее) вычисление соответствует одному моменту времени (одному такту часов). В данной модели отрезок времени между i и $i+1$ тактами называется раундом при синхронных вычислениях. За один раунд протокола каждый процессор может получать сообщения от любого другого процессора, выполнять локальные (внутренние) вычисления и посылать сообщения всем другим процессорам сети. Имеется входная лента «только-для-чтения», которая первоначально содержит строку $\Lambda(k)$ (например вида 1^k), при этом k является параметром безопасности системы. Каждый процессор имеет ленту случайных значений, конфиденциальную рабочую ленту «только-для-чтения» (первоначально содержащую строку Λ), конфиденциальную входную ленту «только-для-чтения» (первоначально содержащую строку x_i), конфиденциальную выходную ленту «только-для-записи» (первоначально содержащую строку Λ) и несколько коммуникационных лент. Между каждой парой процессоров i и j существует конфиденциальный и/или открытый канал связи, посредством которого i посылает сообщение процессору j. Данный канал (коммуникационная лента) исключает чтение для i и исключает запись для j. Каждый процессор i имеет также широковещательный канал, представляющий собой ленту, исключающую чтение для i и являющийся каналом «только-для-чтения» для всех остальных процессоров сети.

Предполагается, что в раунде r для каждого процессора i существует однозначно определенное сообщение (возможно пустое), которое распространяет процессор i в этом раунде и для каждой пары процессоров i и j существует единственное сообщение, которое i может открытым или конфиденциальным образом послать j в данном раунде. Все каналы являются помеченными так, что каждый получатель сообщения может идентифицировать его отправителя.

Процессор i запускает программы π_i, $i = 1, \ldots, n$, совокупность которых реализует распределенный алгоритм $\mathbf{\Pi}$. Протоколом работы сети называется n-элементный кортеж $P = (\pi_1, \pi_2, \ldots, \pi_n)$. Протокол называется t-устойчивым, если t процессоров являются сбоящими во время выполнения протокола. Историей процессора i являются: содержание его конфиденциального и общего входов, все распространяемые им сообщения, все сообщения, полученные i по конфиденциальным каналам связи, и все случайные параметры, сгенерированные процессором i во время работы сети.

3.2. Идеальный и реальный сценарии

Для доказуемо конфиденциального вычисления вводятся понятия идеального и реального сценариев [7]. Как было показано выше в идеальном сценарии (идеальной модели) дополнительно вводится доверенный процессор. Процессоры конфиденциально посылают свои входы доверенному процессору, который вычисляет необходимый результат

(выход) и также конфиденциально посылает его обратно процессорам сети. В случае адаптивного противника, последний может манипулировать с этим результатом (вычислить или изменить его) следующим образом. До начала вычислений он может подкупить один из процессоров и изучить его конфиденциальный вход. Основываясь на этой информации, противник может подкупить второй процессор и изучить его секретный вход. Это продолжается до тех пор пока противник не получит всю необходимую для него информацию. Далее у противника есть два основных пути. Он может изменить входы сбоящих процессоров. После чего те, вместе с корректными входами несбоящих процессоров, направляют свои новые измененные входы TP-процессору. По получению от последнего выходов (значения вычисленной функции) противник может приступить к изучению выхода каждого сбоящего процессора. Второй путь заключается в последовательном изучении входов и выходов процессоров, подключая их всякий раз к числу сбоящих. В данном случае рассматривается противник, который не только может изучать входы сбоящих процессоров, но и менять их, пробовать изучать по полученному значению функции конфиденциальные входы несбоящих процессоров.

В реальном сценарии не существует доверенного процессора, и все процессоры моделируют его поведение посредством выполнения многостороннего интерактивного протокола. Грубо говоря, считается, что вычисления в действительности (в реальном сценарии) безопасны, если эти вычисления «эквивалентны» вычислениям в идеальном сценарии. Точное определение (формальное определение) понятия эквивалентности в этом контексте является одной из основных проблем в теории конфиденциальных вычислений [7].

4. Асинхронная модель вычислений

4.1. Общее описание модели

В данном подразделе рассматривается полностью асинхронная сеть из n процессоров, которые соединены конфиденциальными каналами связи. При этом не существует единых глобальных часов. Любое сообщение в сети может быть задержано независимым образом. В то же время считается, что каждое посланное сообщение обязательно будет получено адресатом. Вопрос о переупорядочивании сообщений не исследуется.

Вычисления в асинхронной модели рассматривается как последовательность шагов. На каждом шаге активизируется один из процессоров. При этом активизация процессора происходит по получении им сообщения. После чего он выполняет внутренние вычисления и возможно выдает сообщения в свои выходные каналы. Порядок шагов определяется планировщиком D, неограниченным в вычислительной мощности. В данной модели вычислений каждый шаг рассматривается как раунд при асинхронных вычислениях.

4.2. Асинхронные идеальный и реальный сценарии

Идеальный сценарий в асинхронной модели с доверенным процессором заключается в добавлении этого процессора в существующую асинхронную сеть при наличии t потенциальных сбоев (сбоящих процессоров). При этом несбоящие процессоры, также как и доверенный процессор, не могут ожидать наличия более чем $n - t$ входов для вычислений с целью получения их выходов, так как t процессоров (сбоящие процессоры) могут никогда не присоединиться к вычислениям.

В начале вычислений процессоры посылают свои входы доверенному процессору. В то же время, существует планировщик D, который доставляет сообщения от процессоров некоторому базовому подмножеству процессоров, мощностью не меньше $n - t$, обозначаемому как C и являющемуся независимым от входов честных процессоров. Доверенный процессор по получению входов — аргументов функции (возможно некорректных) из множества C, предопределенно оценивает значение вычисляемой функции, основываясь на C и входах процессоров из C. (Здесь для корректности может использоваться следующее предопределенное оценивание: установить входы из C в 0 и вычислить данную функцию). Затем доверенный процессор посылает значение оценочной функции обратно процессорам совместно с базовым множеством C. Наконец несбоящие процессоры выдают то, что они получили от доверенного процессора. Сбоящие процессоры выдают значение некоторой независимой функции, информацию о которой они «собирали» в процессе вычислений. Эта информация может состоять из их собственных входов, случайных значений, используемых при вычислениях и значения оценочной функции.

Так же как и в синхронной модели, вычисления в реальной асинхронной модели безопасны, если эти вычисления «эквивалентны» вычислениям в сценарии с доверенным процессором [9].

Заключение

Проактивно безопасная вычислительная система может быть реализована с использованием идей из теории конфиденциальных вычислений. К ним относятся многосторонние протоколы конфиденциально-

го вычисления функции в различных моделях взаимодействия, моделях противника, в синхронных и асинхронных сетях. Значительная часть таких протоколов рассмотрена в работах [1], [2]. К этой разновидности распределенных вычислений можно отнести следующие протоколы, имеющие, в том числе, и прикладное значение:

- протоколы византийских соглашений;
- протоколы проверяемого разделения секрета;
- протоколы электронного голосования;
- протоколы выработки общей случайной строки;
- протоколы неочевидного обмена и многие другие.

Таким образом, методы конфиденциальных вычислений могут позволить для вычислительных систем проектировать высокозащищенную программно-аппаратную среду для использования в автоматизированных системах различных объектов информатизации. Основным принципом создания таких систем является принцип обеспечения проактивной безопасности, который позволяет защищаться от различного рода злоумышленных действий, как на этапе разработки, так и на этапе эксплуатации вычислительных систем.

Список литературы

[1] *Казарин О. В.* Безопасность программного обеспечения компьютерных систем. М.: МГУЛ, 2003, 212 с.

[2] *Казарин О. В.* Теория и практика защиты программ. 2004, 450 с. `http://www.cryptography.ru`.

[3] *Казарин О. В.* О создании информационных технологий, исходно ориентированных на разработку безопасного программного обеспечения. Вопросы защиты информации. 1997. № 1–2 (36–37). С. 12–14.

[4] *Yao A. C.* Protocols for secure computations (extended abstract). Proc. 23rd IEEE Symp. on Foundations of Computer Science (FOCS'82). 1982. P. 160–164.

[5] *Yao A. C.* How to generate and exchange secrets. Proc. 27th IEEE Symp. on Foundations of Computer Science (FOCS'86). P. 162–167.

[6] *Goldreich O.*, *Micali S.*, *Wigderson A.* How to play any mental game. Proc. 19th ACM Symposium on Theory of Computing (STOC'87). P. 218–229.

[7] *Micali S.*, *Rogaway Ph.* Secure computation. Lecture Notes in Computer Science. Advances in Cryptology — CRYPTO'91. V. 576. P. 392–404.

[8] *Beaver D.* Foundations of secure computing. Lecture Notes in Computer Science. Advances in Cryptology — CRYPTO'91. V. 576. P. 377–391.

[9] *Ben-Or M.*, *Canetti R.*, *Goldreich O.* Asynchronous secure computation. Proc. 25th Annual ACM Symposium of Computing (STOC'93). P. 52–61.

Защищенная гибридная операционная система «Linux over Феникс»

Д. П. Зегжда, А. М. Вовк

В статье рассмотрен подход к созданию защищенных систем обработки информации, в основе которого лежит технология гибридных операционных систем, позволяющая одновременно использовать несколько различных операционных систем. В качестве базовой предлагается использовать защищенную ОС Феникс, реализующую функции защиты, а в качестве встраиваемой — популярную ОС Linux, обеспечивающую совместимость с распространенными приложениями.

1. Технология гибридных систем

Суть технологии гибридных операционных систем заключается в том, что в рамках одной операционной системы (ее называют «базовой») создается среда, в которой может функционировать другая операционная система (ее называют «встроенной»).

Данная технология может быть применена для построения защищенной операционной системы, в которой базовая защищенная ОС обеспечивала бы безопасность, а встраиваемая ОС — совместимость с приложения и пользовательский интерфейс. Для этого необходимо чтобы встраиваемая ОС могла быть запущена как обычный пользовательский процесс в составе защищенной ОС. Так же необходимы средства, которые обеспечат приложениям встраиваемой ОС доступ к ресурсам защищенной ОС. С их помощью приложения встраиваемой ОС будут обращаться к ресурсам защищенной ОС под контролем реализованных в ней средств защиты. Таким образом множество приложений защищенной ОС расширяется за счет уже существующих и разрабатываемых приложений встраиваемой ОС.

Примером такой защищенной гибридной системы является решение, разработанное на кафедре информационной безопасности компьютерных систем факультета технической кибернетики СПбГПУ, получившее название защищенная гибридная система «Linux over Феникс». В каче-

стве защищенной ОС, имеющей специальную архитектуру и реализующей гибкую модель управления доступом к информационным ресурсам выступает ЗОС Феникс. В среде этой ОС на правах обычного пользовательского процесса выполняются экземпляры модифицированного ядра Linux, адаптированного для работы в пользовательском режиме в среде ЗОС Феникс. Каждый пользователь имеет в своем распоряжении персональную копию среды Linux, полностью изолированную от всех остальных. Для доступа к ресурсам защищенной ОС Феникс используется драйвер файловой системы модифицированного ядра Linux, который перенаправляет обращения к ресурсам Феникс, которые находятся под контролем средств защиты ЗОС Феникс.

Данное решение позволяет расширить множество приложений, запускаемых под ЗОС Феникс широким множеством приложений популярной ОС Linux. Таким образом все функции защиты реализованы в составе Феникс, а для прикладных процессов открыты все функциональные возможности Linux. Причем при этом не наносится ущерб защищенности ОС Феникс, а все приложения Linux подчиняются модели безопасности Феникс.

2. Виртуальная машина Феникс для Linux

ОС Linux, функционирующая в среде Феникс представляет собой обычный процесс ОС Феникс, в котором размещаются: виртуальная машина Феникс для Linux, модифицированное ядро Linux и пользовательские процессы Linux.

Виртуальная машина Феникс для Linux включает в себя:

- модуль управления памятью, который позволяет отображать на заданный виртуальный адрес заданную физическую страницу;
- модуль управления исключительными ситуациями и прерываниями, который позволяет обрабатывать исключительные ситуации процессора и прерывания в пользовательском режиме виртуальной машины Linux.

В каждый момент времени в виртуальном адресном пространстве виртуальной машины находятся страницы ядра ОС Linux и страницы текущего пользовательского процесса Linux. Для каждого процесса ОС Linux ведется список используемых страниц, который модифицируется по мере выделения/освобождения памяти. Когда квант времени, который отведен текущему процессу Linux, истекает, ядро Linux удаляет из виртуального адресного пространства страницы, принадлежащие вытесняемому процессу, и отображает на их место страницы, принадлежащие процессу, на который осуществляется переключение.

В ходе функционирования виртуальной машины ей необходимо обрабатывать некоторые исключительные ситуации процессора и прерывания (например, прерывание от таймера или страничные ошибки). Когда возникает исключительная ситуация или прерывание, относящееся к процессу виртуальной машины, ядро Феникс передает ей управление. В случае, если виртуальная машина некорректно обработает прерывание или исключительную ситуацию, целостность и стабильность ОС Феникс не нарушается — процесс этой машины завершится.

3. Безопасность

Linux в среде Феникс функционирует в пользовательском режиме, как обычный процесс ОС Феникс. Это значит, что Linux не может нарушить работу ядра ОС Феникс и других приложений, выполняющихся в среде Феникс.

Более того, виртуальная машина Феникс для Linux с помощью механизмов границ сегментов, уровней привилегий и виртуальной памяти защищает ядро Linux от пользовательских процессов Linux и пользовательские процессы Linux друг от друга. Для решения этой задачи используются возможности по защите памяти процессоров семейства IA-32.

1. *Защита ядра Феникс от ядра Linux*. Ядро Феникс защищено от ядра Linux при помощи страничной организации памяти (ядро Феникс выполняется в привилегированных страницах, ядро Linux — в пользовательских страницах). Виртуальная машина Linux является обычным процессом ОС Феникс и не имеет доступа к внутренним структурам ядра ОС Феникс.

2. *Защита виртуальной машины Linux от процессов Linux*. Виртуальная машина Linux защищена от процессов Linux при помощи сегментной организации памяти и уровней привилегий сегментов.

3. *Защита процессов Linux друг от друга*. Младшие три гигабайта виртуального адресного пространства виртуальной машины используются для работы процессов Linux. Перед переключением с одного процесса на другой память, используемая первым процессом, удаляется из виртуального адресного пространства. Память, используемая новым процессом, отображается в виртуальное адресное пространство, только после чего управление передается новому процессу. Таким образом, процессы ОС Linux имеют доступ только к своему адресному пространству, но не имеют доступа к виртуальным адресным пространствам других процессов ОС Linux и, следовательно, не могут влиять на их работу.

4. *Защита ядра Феникс от процессов Linux*. Процессы ОС Linux выполняются на 3-м уровне привилегий в сегменте, ограниченном тремя гигабайтами, и не имеют доступа к внутренним структурам ядра Феникс, и, следовательно, они не могут нарушить его работу или повлиять на другие процессы Феникс.

Таким образом, компоненты гибридной системы образуют иерархию: «ядро Феникс» — «виртуальная машина Linux» — «процесс Linux», в которой каждый компонент полностью контролирует компоненты нижнего уровня и защищен от их влияния.

4. Совместимость

Виртуальная машина Феникс для Linux имеет высокий уровень совместимости с оригинальной ОС Linux. Это достигается тем, что виртуальная машина Феникс для Linux не пытается повторить или эмулировать функциональные возможности Linux, а представляет собой стандартное ядро ОС Linux версии 2.4, в которое внесены небольшие изменения, позволяющие запускать его как обычный процесс Феникс. Эти изменения касаются небольшого количества модулей и могут быть легко повторены со следующими версиями ядра Linux.

5. Производительность

Приложения Linux, выполняющиеся в виртуальной машине Феникс для Linux, обладают примерно таким же уровнем производительности, что и при работе в оригинальной ОС Linux. Этого удается достичь благодаря тому, что выполнение системных вызовов в ядро Linux не эмулируется, а выполняются также, как и в оригинальной ОС Linux. Существуют накладные расходы, связанные с контролем безопасности со стороны Феникс, но они проявляются лишь при попытке доступа к ресурсам Феникс, а не постоянно во время работы приложения Linux. Причем эти дополнительные накладные расходы будут не больше, чем если бы данное приложение портировалось в среду Феникс.

6. Преимущества технологии гибридных операционных систем

Использование технологии гибридных операционных систем при построении защищенных информационных систем на базе ЗОС Феникс и популярной открытой ОС Linux позволяет обеспечивать следующие свойства:

- тотальный контроль всех информационных взаимодействий и потоков информации со стороны доверенных средств защиты из состава ЗОС Феникс, что обеспечивает высокий уровень безопасности.
- невозможность обхода или отключения защиты, поскольку средства защиты Феникс взаимодействуют непосредственно с аппаратной платформой, а средства Linux, напротив, не имеют к ней доступа.
- множество доступных приложений расширяется за счет приложений Linux, что позволяет использовать гибридную систему «Linux over Феникс» практически везде, где применяется Linux.
- минимизация накладных расходов на организацию защиты — фактически единственным дополнительно исполняемым кодом по сравнению с обычной Linux является код средств защиты ЗОС Феникс.

О некоторых задачах анализа устойчивости работы информационных сетей

В. К. Попков, О. Д. Соколова, А. Н. Юргенсон

Введение

В работе современных систем управления сетями одним из важных аспектов является сетевой мониторинг. Параметрами для исследования работы сети могут быть статус устройств, их производительность, число пакетов, изменение топологии и т. д. Чем сложнее топология и конфигурация сети, тем больше информации требуется для ее адекватного представления и анализа [1]. В связи с этим моделирование информационной сети должно давать наглядные объекты, удобные для решения задач мониторинга. Представление сети в виде графа позволяет решать узкий круг задач, поэтому для более адекватного представления принято использовать гиперсетевые структуры — абстрактные гиперсети, иерархические гиперсети и др. [2].

1. Анализ работы сети в интерактивной среде

Механизмом защиты, который может выявлять вторжения в сеть в режиме реального времени, являются так называемые системы обнаружения вторжений (СОВ). Имеется два основных метода выявления вторжений: выявление некорректного использования (misuse detection) и выявление аномалий. Для выявления некорректного использования применяют сигнатуры известных атак, т. е. образцы атакующего поведения или атакующих действий, определяя соответствующую активность как вторжение. При выявлении аномалий используют установленные нормальные профили, определяя любые неожиданные отклонения, как возможный результат вторжения.

Для характеризации нормальной и аномальной работы информационной системы осуществляется мониторинг различных ее характеристик, например, информационных потоков. Для этого обычно определяются пороги (нижний и верхний) для каждого наблюдаемого параметра сети. Иногда используют и статистическую модель для вычисления вероятности появления данной величины.

Для мониторинга информационных потоков в сети наиболее удобной представляется интерактивная среда, позволяющая в реальном режиме не только исследовать модель сети, но и корректировать ее.

Основными элементами модели сети связи являются модели структуры первичной и вторичной сетей, модели узлов и соединительных линий. Основными элементами модели компьютерной сети являются модель структуры сети, модели хостов, модель вычисления вероятностей успеха действий и модель реакции хоста. Модель структуры сети определяет ключевые параметры сети в целом. Модели хостов задают параметры отдельных хостов (IP-адрес, тип и версия ОС, идентификаторы пользователей и т. п.) [3].

Гиперсеть $S = (X, V, R)$ включает в себя множество вершин, ветвей ребер, которые определяют структуры первичной (X, V) и вторичной (X, R) сетей. Анализ их параметров, мониторинг изменений во времени дает возможность решать некоторые задачи оптимального управлений работой сети.

Для динамического отображения состояния как первичной, так и вторичной сети удобно использовать графический редактор [4]. В качестве модели сети Internet в граф-редакторе можно построить гиперсеть, где первичная сеть будет иметь древовидную структуру, а каналы между парами абонентов образуют вторичную сеть. Работа в редакторе позволяет в интерактивном режиме отслеживать компьютеры, работающие в сети, и потоки передаваемой информации.

2. Моделирование информационных атак

Под атакой будем понимать совокупность действий, приводящих к нарушению информационной безопасности сети. Результатом успешной атаки может стать, например, несанкционированный доступ к информации, хранящейся в сети, потеря работоспособности системы или искажение данных в сети. В качестве целей атаки могут рассматриваться серверы, рабочие станции пользователей или коммуникационное оборудование [5].

Интерактивный режим позволяет показать развитие атаки во времени и изменение вследствие этого характеристик сети. Особенно это важно когда поведение атакующего меняется в соответствии с реакцией сети на атаку.

Для задач мониторинга сети в качестве параметров состояния сети могут выступать уровень загрузки процессора, нагрузка на каналы

связи, штатное время работы пользователей системы, количество обращений к сетевым ресурсам и т. д. [5].

Нагрузка на каналы является одним из универсальных параметров состояния сети. В качестве показателя нагрузки можно взять максимальный поток между выделенной парой вершин.

Для моделирования атак на сеть рассмотрим нестационарную гиперсеть $AS(t) = (X, V, R)$ [2], где каждой ветви v_j и каждому ребру r_i сопоставлены функции пропускной способности $\alpha_j(t)$ и $\delta_i(t)$ соответственно, зависящие от времени.

Рассмотрим задачу моделирования максимального потока из вершины x_1 в вершину x_n во вторичной сети при условии, что пропускная способность ветвей первичной сети изменяется в результате атак на нее [6]. Цель моделирования — идентифицировать атаку на основе изменения максимального потока в сети. Сравним два типа атак на первичную сеть: атака типа «червь» и атака типа «взрыв».

Распространение атаки типа «червь»

1. Пусть атака начата в вершине 1, тогда $k := 1$;
2. Случайным образом выбирается вершина p, смежная с k, в которую переходит «червь». Пропускная способность $\alpha_j(t)$ атакуемой ветви j (соединяющей вершины k и p) первичной сети уменьшается на некоторую постоянную величину (в зависимости от типа атаки).
3. $k := p$; если поток достиг заданного критического значения, то конец, иначе возвращаемся на шаг 2.

Распространение атаки типа «взрыв»

1. Пусть атака начата в вершине 1, тогда $K = \{1\}$ — множество атакованных вершин;
2. $\forall\, k \in K$ во все, смежные с k, вершины переходит атака. Пропускная способность $\alpha_j(t)$ каждой атакуемой ветви j (инцидентной вершине k) первичной сети уменьшается на некоторую постоянную величину (в зависимости от атаки). $K := \varnothing$. В множество K записываются вершины, в которые перешла атака.
3. Если поток достиг заданного критического значения, то конец, иначе возвращаемся на шаг 2.

На рис. 1, 2 представлены результаты моделирования поведения максимального потока при атаках трех типов: червь, забирающий у ветви ресурс в 3 единицы; две атаки типа взрыв, забирающие у ветви ресурс в 1 единицу (первая атака) и 0.1 единицы (вторая атака).

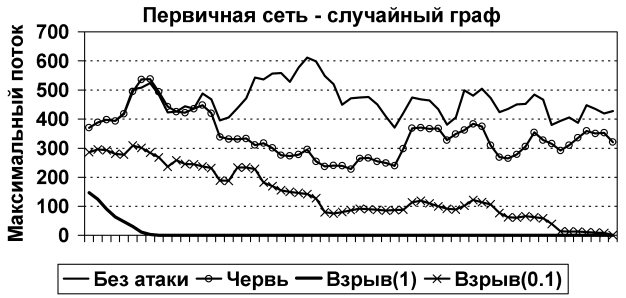


Рис. 1. Поведение максимального потока при атаках. Сеть решетчатой структуры.

Для сравнения приведен график максимального потока во времени при нормальном функционировании сети (без атаки). Моделирование производилось на гиперсетях со следующими параметрами: $|X| = 20$, $|V| = 62$, $|R| = 500$. Рассматривалось два варианта первичной сети — случайный граф и решетка. Вторичная сеть во обоих случаях — случайный граф. В нулевой момент времени атакована некоторая случайная вершина, атака распространяется в каждый момент времени. На графиках отражено значение максимального потока за интервал времени $[t,\ t + 10]$ $\forall t \in (0,\ 50)$. Из рис. 1 видно, что случайные структуры сетей не могут быть адекватными моделями реальных сетей, так как их характеристики имеют очень большую дисперсию при проведении экс-

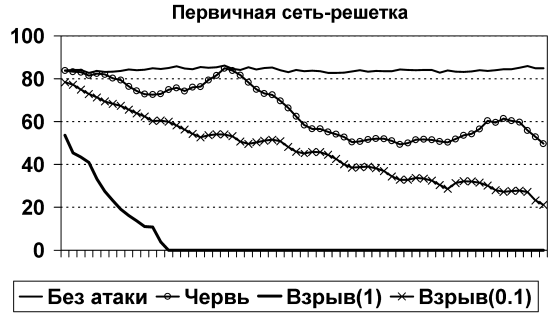


Рис. 2. Поведение максимального потока при атаках. Сеть случайной структуры.

периментов по анализу устойчивости работы сетей к разрушающим информационным воздействиям.

3. О решении некоторых задач с применением гиперсетей

Опишем сведение некоторых задач анализа сетей к задачам поиска покрывающих множеств в гиперсетях [7].

1. Размещение оконечных пунктов циркулярной связи. Так как в сети циркулярной связи системы оповещения располагаются в местах нахождения каналообразующей аппаратуры, то, очевидно, любой оконечный пункт принадлежит вторичной сети. В то же время оповещаемые пункты подсоединены (на прием) к одному или нескольким групповым каналам. Необходимо в этих условиях так разместить минимальное число оконечных пунктов, чтобы все абоненты были подсоединены к групповым каналам, исходящим из оконечных пунктов. Такой постановке соответствует задача поиска минимального множества вершин вторичной сети, покрывающего все вершины первичной сети.

2. Пусть задана сеть электросвязи, линии передачи и коммутационные узлы ненадежны. Для гиперсети $S = (X, V, R)$, где X — множество вершин, V — множество ветвей первичной сети, R — множество ребер вторичной сети требуется узнать, какое минимально возможное число линий передач и узлов коммутации надо вывести из строя, чтобы отказали все каналы. Это задача поиска минимального множества из X и V, покрывающего все элементы множества R.

3. Построение сети ретрансляторов. Пусть X — населенные пункты некоторой территории, а V — районы. Предположим, что для различных подмножеств районов доступны определенные каналы с различными программами. Необходимо найти места расположения минимального числа телетрансляторов так, чтобы были покрыты все районы и доступны все каналы. Задача в терминах гиперсетей имеет вид: поиск минимального множества из X, покрывающего V и R.

Заключение

Рассматриваемые математические модели (гиперсети, нестационарные гиперсети и др.) и разработанные в связи с актуальными задачами мониторинга сетей методы позволяют не только анализировать живучесть различных сетей, но и решать задачи моделирования сетей с заданными воздействиями, структурными требованиями и ограничениями.

В настоящее время эти методы могут быть использованы для анализа современных информационных сетей.

Список литературы

[1] *Бредихин С. В.*, *Ляпунов В. М.*, *Щербакова Н. Г.* Сетевой мониторинг. Обзор и опыт применения. Труды ИВМиМГ, Информатика, 4, Новосибирск, 2002, стр. 25–35.

[2] *Попков В. К.* Математические модели связности. Новосибирск, 2002.

[3] *Городецкий В. И.*, *Котенко И. В.* Командная работа агентов-хакеров: применение многоагентной технологии для моделирования распределенных атак на компьютерные сети. `http://space.iias.spb.su/ai/doc/CAI-2002-2.pdf`.

[4] *Зыков М. Л.*, *Соколова О. Д.* Интерактивная система анализа и синтеза проектных решений в сетях электросвязи. Труды ИВМиМГ, Информатика, т. 4, Новосибирск, 2002, с. 175–179.

[5] *Сердюк В.* Вы атакованы — защищайтесь! `http://www.bytemag.ru/Article.asp?ID=2030`.

[6] *Попков В. К.*, *Соколова О. Д.*, *Юргенсон А. Н.* К вопросу о моделировании информационных атак с помощью гиперсетей. Proceedings of 8th International Conference «Problems of Operation of Information Networks» (Bishkek, 22–29 August 2004), Новосибирск, т. 2, стр. 258–263.

[7] *Popkov V. K.*, *Sokolova O. D.* Covering in Hypernet. CSIT'2000, Ufa, Russia, p. 12–13.

Использование метода ветвления для точного расчета вероятности связности случайного графа

А. С. Родионов, О. К. Родионова, Д. А. Мигов, М. Ю. Мурзин

Введение

В ходе атак на информационные сети, а также по их завершению, как результат атаки, возможно нарушение отдельных связей и узлов сети, приводящее к изменению характеристик связности. Полученная оценка этих характеристик имеет важное значение для оценки возможного ущерба от разрушающих воздействий различного рода. Существует много характеристик связности и производительности сетей. Мы рассмотрим такую важную характеристику, как вероятность ее связности.

В качестве модели рассматривается случайный граф. Возможно рассмотрение различных вариантов: надежные вершины и ненадежные ребра, надежные ребра и ненадежные вершины, ненадежные вершины и ребра, рассмотрение тотальной связности (каждая из оставшихся вершин связана с каждой) и связности заданного подмножества «целевых» вершин и т. д. Исследования в данной области проводились многими авторами [1]–[8], однако все они, за исключением А. Шумана (A. Shooman) констатировали невозможность применения точных методов расчета вероятности связности в силу экспоненциальной сложности задачи (в дальнейшем, для краткости, эту вероятность будем называть *надежностью графа*). Однако мощности современных ЭВМ, совместно с некоторыми алгоритмическими приемами, позволяют применять точные методы расчета для сетей с относительно небольшой, но практически интересной размерностью (до сотен элементов). В настоящей статье в основном ограничимся рассмотрением графов с абсолютно надежными вершинами и ненадежными ребрами. Дальнейшее изложение организовано следующим образом: во втором разделе изложены основы метода ветвления и способ его ускорения за счет

использования простых цепей, в третьем разделе приведены приемы редукции размерности графа и его декомпозиции, в четвертом — некоторые вопросы программной реализации метода. В последнем, пятом, разделе приводятся результаты сравнительных экспериментов и предлагаются направления дальнейших работ.

1. Метод ветвления

Широко известный метод ветвления (Мура — Шеннона) заключается в рекурсивном применении формулы полной вероятности при рассмотрении в качестве альтернативных гипотез наличия либо отсутствия очередного разрешающего элемента. При рассмотрении случая абсолютно надежных вершин и ненадежных ребер формула принимает следующий вид [1] (реализация описана в [9]):

$$R(G) = \varepsilon_{ij} R(G^*(e_{ij})) + (1 - \varepsilon_{ij}) R(G \setminus \{e_{ij}\}), \tag{1}$$

где $G^*(e_{ij})$ — граф, стянутый по ребру e_{ij}, имеющему вероятность присутствия (далее *надежность*) ε_{ij}, $G \setminus \{e_{ij}\}$ — граф, получаемый из G удалением ребра e_{ij}.

В случае ненадежных вершин и абсолютно надежных ребер имеем

$$R(G) = \varepsilon_i R(G^*(v_i)) + (1 - \varepsilon_i) R(G \setminus \{v_i\}), \tag{2}$$

где $G^*(v_i)$ — граф, в котором стянуты все ребра e_{ij}, инцидентные вершине v_i, имеющей надежность ε_i, $G \setminus \{v_i\}$ — граф, получаемый из G удалением вершины v_i.

Рекурсии продолжаются до получения либо несвязного графа (возвращается 0), либо до получения графа, для которого надежность можно вычислить непосредственно. Очевиден быстрый экспоненциальный рост числа рекурсий с ростом размерности графа.

Далее, если не оговорено противное, рассматривается случай абсолютно надежных вершин и ненадежных ребер.

В [10, 12, 13] предложена модификация метода ветвления для случая абсолютно надежных вершин и ненадежных ребер, позволяющая сократить число рекурсий при наличии в графе простых цепей, т. е. цепей, проходящих по вершинам степени 2. Модификация определяется следующей теоремой.

Теорема 1. *Пусть граф G содержит простую цепь из k ребер $e_1, e_2, \ldots, e_k$ с вероятностями присутствия $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$ и соединяющую вершины s и t. Тогда надежность графа G равна*

$$R(G) = \prod_{j=1}^{k} \varepsilon_j \cdot R(G^*(e_1, \ldots, e_k)) + \sum_{i=1}^{k} (1-\varepsilon_i) \prod_{j\neq i} \varepsilon_j \cdot R(G \setminus \{e_1, \ldots, e_k\}) \tag{3}$$

при отсутствии ребра e_{st} *и*

$$\begin{aligned} R(G) = {} & \left[(\varepsilon_1 + \varepsilon_{st} - \varepsilon_1\varepsilon_{st}) \prod_{j=2}^{k} \varepsilon_j + \varepsilon_{st} \sum_{i=2}^{k} (1-\varepsilon_i) \prod_{j\neq i} \varepsilon_j \right] \times \\ & \times R(G^*(e_1, \ldots, e_k)) \\ & + \left[(1 - \varepsilon_1 - \varepsilon_{st} + \varepsilon_1\varepsilon_{st}) \prod_{j=2}^{k} \varepsilon_j + (1-\varepsilon_{st}) \sum_{i=2}^{k} (1-\varepsilon_i) \prod_{j\neq i} \varepsilon_j \right] \times \\ & \times R(G \setminus \{e_1, \ldots, e_k, e_{st}\}) \end{aligned} \tag{4}$$

при его наличии.

Доказательство теоремы основано на последовательном применении формулы (1) к ребрам e_k, e_{k-1}, …, e_1 (и, возможно, ребру e_{st}), и последующим приведении подобных. При этом используется тот факт, что если граф G имеет висящую вершину, прикрепленную к остальной части графа ребром e, имеющим надежность ε, то $R(G) = \varepsilon \cdot R(G \setminus \{e\})$.

Отметим, что даже если в исходном графе таких цепей нет, они могут появиться в ходе рекурсивного ветвления. Удаление ребра e_{ij} приводит к уменьшению степеней вершин i и j на 1, и, если разрушение графа не произойдет раньше, степень какой-либо вершины неизбежно достигнет 2, т. е. в графе появится простая цепь. Стягивание пары вершин также может привести к появлению вершин степени 2.

2. Снижение размерности задачи

Очевидные приемы снижения размерности задачи связаны с выделением компонент графа, соединенных мостами либо точками сочленения, а также удалением висячих вершин [11]. Существенный эффект, как на предварительном этапе, так и в ходе вычислений, может дать последовательно-параллельная редукция ребер.

Если два графа G_1 и G_2 имеют единственную общую вершину (точку сочленения), то надежность объединенного графа G есть

$$R(G) = R(G_1) \cdot R(G_2). \tag{5}$$

Естественным следствием этого является то, что надежность графа G, полученного из графов G_1 и G_2 соединением цепью из k ребер

$e_1, e_2, \ldots, e_k$ с вероятностями присутствия $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$, будет равна

$$R(G) = \prod_{i=1}^{k} \varepsilon_i \cdot R(G_1) \cdot R(G_2). \tag{6}$$

Из равенства (5) следует и то, что надежность произвольного дерева равна произведению вероятностей присутствия входящих в него ребер. Отсюда простой способ редукции графа перед применением метода вычисления его надежности: из него удаляются все ребра, входящие в «прикрепленные» деревья, вместе с вершинами, кроме вершин, к которым эти деревья «прикреплены». Надежность графа есть произведение вероятностей присутствия всех удаленных ребер на надежность оставшегося (редуцированного) графа. Далее предполагаем, что минимальная степень вершины графа, представленного к расчету, не менее 2.

Мосты и точки сочленения возможно выявить на предварительном этапе, например, визуальным анализом или применением соответствующих алгоритмов поиска. В ходе расчета надежности рассматриваемым ниже методом специальный поиск этих структурных особенностей нецелесообразен, но они должны учитываться, если получаются по ходу решения.

2.1. Редукция цепей

В [5] предложено заменять пару ребер с надежностями ε_1 и ε_2, имеющих общую вершину степени 2, на одно ребро таким образом, чтобы вероятность связности всех оставшихся вершин графа, умноженная на некоторый однозначно вычисляемый фактор, равнялась вероятности связности всех вершин исходного графа. При этом вероятность присутствия ребра, заменяющего цепь длины 2, равна

$$p = \frac{p_1 p_2}{1 - (1-p_1)(1-p_2)} = \frac{p_1 p_2}{p_1 + p_2 - p_1 p_2}, \tag{7}$$

а коэффициент (фактор)

$$r = p_1 + p_2 - p_1 p_2. \tag{8}$$

Основываясь на этом результате и рассматривая последовательную редукцию пар ребер для цепи длины $k > 2$, можно сформулировать следующую теорему [13], [14]:

Теорема 2. *Пусть граф* $G_1(n, m)$ *содержит простую цепь из* k *ребер* $e_1, e_2, \ldots, e_k$ *с вероятностями присутствия* $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$.

Тогда надежность графа $G_1(n, m)$ *равна*

$$R\big(G_1(n, m)\big) = \prod_{i=1}^{k} \varepsilon_i \left(\sum_{i=1}^{k} \varepsilon_i{}^{-1} - k + 1 \right) R\big(G_2(n-k+1, m-k+1)\big), \tag{9}$$

где граф $G_2(n-k+1, m-k+1)$ *получен из* $G_1(n, m)$ *заменой этой цепи ребром с вероятностью присутствия*

$$\varepsilon = \left(\sum_{i=1}^{k} \varepsilon_i^{-1} - k + 1 \right)^{-1}. \tag{10}$$

2.2. Использование вершинных разрезов

В работе [15] доказана следующая теорема, позволяющая снижать размерность задачи за счет декомпозиции графа, имеющего вершинный разрез мощности 2:

Теорема 3. *Пусть граф* G *состоит из двух компонентов* G_1 *и* G_2, *имеющих ровно две общие вершины* x *и* y. *Тогда справедливо*

$$R(G) = R(G_1)R(G_2) + R(G_1)[R(G_2') - R(G_2)] + R(G_2)[R(G_1') - R(G_1)], \tag{11}$$

где G_i' *есть граф* G_i, *в котором вершины* x *и* y *стянуты в одну*.

Применяя формулы ветвления по ребрам или цепям, всегда начинаем с шага удаления, поскольку результат может показать наличие моста и тогда по линии стягивания возможно вычислять надежность по формуле (6).

3. Оконечные рассчитываемые варианты графов

При расчете надежности графа по фомулам (1), (3) и (4) в процессе редукции рано или поздно получаются графы, для которых возможно непосредственно получить надежность:

1. *Несвязный граф*. Вероятность связности равна нулю.

2. *Цикл*. Вероятность связности цикла, состоящего из k ребер $e_1, \ldots, e_k$ с вероятностями присутствия $\varepsilon_1, \ldots, \varepsilon_k$, есть

$$R(G) = \prod_{i=1}^{k} \varepsilon_i \left[1 + \sum_{i=1}^{k} \frac{1-\varepsilon_i}{\varepsilon_i} \right]. \tag{12}$$

3. *Граф малой размерности*. Простейшим вариантом является граф из одного ребра, процедура должна вернуть вероятность его присутствия. Однако желательно рассчитывать надежность напрямую для возможно бóльшей размерности графа, так как это снижает число рекурсивных вызовов процедуры. При этом необходимо организовать вычисления, используя минимальное число операций, так как число оконечных графов может достигать многих и многих миллионов и миллиардов. В [13] приводится формула для эффективного вычисления надежности 4-х вершинного графа, в [16] дается ее улучшение и приводятся формулы для расчета надежности 5-вершинного графа.

4. *Дерево*. Легко проверяемый вариант (граф связен и число ребер на единицу меньше числа вершин), который довольно часто возникает при применении простой формулы Мура — Шеннона и позволяет существенно сократить число рекурсий. Он никогда не возникает в предлагаемом алгоритме, так как висячие вершины отрабатываются немедленно при обнаружении. Для дерева, очевидно, вероятность связности равна произведению вероятностей присутствия всех его ребер.

4. Программная реализация методов

Как и в большинстве случаев реализации численных методов, правильный выбор формата представления данных и эффективность кодирования алгоритма существенно влияют на быстродействие программ и требования к оперативной памяти. Так, использование для представления случайного графа матрицы вероятностей $P = \|\varepsilon_{ij}\|$ кажется избыточным, особенно при малой средней степени вершины. Кроме того, поиск ненулевых элементов (что нужно, например, при вычислении надежности кольца) требует просмотра всех над или поддиагональных элементов матрицы, т. е. имеет квадратичную сложность. Однако именно матрица вероятностей позволяет переиспользование одной и той же области памяти при реализации метода ветвления на последовательной машине, как показано в [9], [12]. Вместе с тем, использование списка ребер позволяет минимизировать объем передаваемой между процессами информации при параллельной реализации метода.

Выше уже отмечалась необходимость при ветвлении сначала производить удаление ребра (цепи) и лишь затем, если полученный таким образом граф связен, производить стягивание вершин. Стоит обратить внимание на то, что при реализации процедуры проверки графа на связность методом поиска в глубину нужно, кроме булевского признака связности, возвращать и список помеченных в ходе проверки вершин (V_1). Если проверка показывает несвязность графа, то он, очевидно, состоит из двух компонентов, имеющих множества вершин V_1 и $V_2 = V \setminus V_1$ в случае удаления ребра и V_1 и $V_2 = V \setminus V_1 \setminus V(C)$ при удалении цепи C, где $V(C)$ — множество ее вершин (кроме оконеч-

ных). Тем самым нет нужды в специальной процедуре разделения вершин по компонентам связности.

Необходимость редукции цепи либо ветвления по ней возникает при наличии в графе вершин степени 2. Очевидно, более эффективно отслеживать степени вершин в ходе ветвления, чем рассчитывать их на каждом шаге. При удалении ребра e_{ij} либо цепи степени оконечных вершин (v_i и v_j) уменьшаются на 1, в то время как при стягивании степень объединенной вершины равна $\deg(v_i) + \deg(v_i) - k + 1$, где k — число кратных ребер, получившихся при стягивании (не забудем, что кратные ребра при расчете заменяются одним с эквивалентной вероятностью присутствия).

В случае рассмотрения графов с ненадежными вершинами и абсолютно надежными ребрами можно упростить программирование метода (сложность при программировании ветвлениия по формуле (2) представляет проведение стягивания) можно воспользоваться приемом из [17]. По ветви стягивания предлагается просто установить надежность разрешающей вершины равной единице. В дальнейшем абсолютно надежные вершины в процессе ветвления участия не принимают. По достижении графа, в котором все вершины абсолютно надежны, в качестве его надежности возвращается единица. Разумеется, применение этого приема требует несколько больших затрат памяти и передачи бóльшего объема информации при параллельной реализации, но оно существенно упрощает программирование и сокращает число операций. Кроме того, этот же прием позволяет просто организовать вычисление надежности графа с ненадежными вершинами и ребрами: начальное ветвление идет по вершинам, по достижении графа с абсолютно надежными вершинами он передается на процедуру вычисления графа с абсолютно надежными вершинами и ненадежными ребрами.

5. Результаты экспериментов

На основе изложенного выше материала реализованы программы расчета вероятности связности случайного графа с абсолютно надежными вершинами и ненадежными ребрами и с абсолютно надежными ребрами и ненадежными вершинами.

Проведенные эксперименты показали много более высокую эффективность предложенных методов как по сравнению с классической формулой Мура — Шеннона, так и по сравнению с методом, предложенным в [2]. Так, время, затраченное на расчет решетки 4 на 4 на компьютере с процессором Pentium III 800 MHz, составило 47 с для метода из [2] (при том, что оригинальный алгоритм был улучшен за счет более ранних завершений рекурсий при достижении графом размерности не более 4 вершин), тогда как программа, реализующая предложенное расширение формулы Мура — Шеннона с учетом всех возможных вариантов оконечных графов, потребовала всего 0.22 с на этом же примере. Базовый алгоритм Мура — Шеннона потребовал 28 с, а дополненный редукцией цепей и учетом висячих вершин на каждом шаге ветвления — всего 0.16 с. Использование вершинных разрезов дало минимальное время в 0.08 с. Более подробные результаты по затратам на расчет надежности случайных графов с абсолютно надежными вершинами и ненадежными ребрами представлены в таблице 1. Здесь под кольцом $n \times m$ понимается граф, представляющий собою циклическое соединение n циклов по m ребер такое, что каждый малый цикл соединяется с соседним одним ребром и вершины для соединения в малых циклах выбираются отстоящими на максимальное расстояние (циклы делятся пополам). Время счета приведено в секундах. Использовался компьютер на базе Intel Celeron 1800 MHz.

Граф	Разм.	Мур — Шеннон		Цепи	
		рекурсии	время	рекурсии	время
полный	8	4319	5	1433	3
полный	9	35279	47	10868	24
полный	10	322559	489	92763	232
решетка 3×3	9	88	0.06	8	0.01
решетка 4×4	16	21263	15	303	0.4
решетка 5×5	25	>8700000	>30 мин.	149147	157
кольцо 4×3	12	234	0.16	15	0.02
кольцо 4×4	16	1071	0.8	39	0.05
кольцо 4×5	20	4338	3	97	0.16
кольцо 4×6	24	16407	12	230	0.4
кольцо 6×4	24	20222	16	67	0.12
кольцо 6×5	30	212776	184	208	0.36

Таблица 1. Вычислительные затраты базового метода ветвления с и без редукции цепей.

Интересным дополнительным примером является расчет надежности графа сети ARPANET с 58 вершинами и 71 ребром. При использовании базового метода ветвления расчет не завершился за сутки. При использовании метода ветвления с редукцией цепей потребовалось 7705 рекурсий и 0.22 с, а при дополнительной проверке на на-

личие вершинного разреза мощности 2 и использовании формулы (11) 7359 рекурсий и 0.17 с соответственно.

Параллельная реализация ветвления по цепям [18] на кластере из 10 вычислителей показала 8-кратное ускорение на 12-вершинных случайных графах, что еще более повышает возможности предложенных методов для расчета графов размерности, соответствующей потребностям анализа локальных информационно-вычислительных сетей.

Дальнейшие исследования предполагается проводить в области поиска методов направленного перебора при ветвлении и в области разработки приближенных методов расчета, основанных на прекращении ветвления на определенной глубине и оценке надежности нерассчитанных вариантов. Большие перспективы по повышению размерности рассчитываемых графов лежат в области разработки эффективных стратегий распараллеливания вычислений.

Список литературы

[1] *Мур Э.*, *Шеннон К.* Надежные схемы из ненадежных реле. Кибернетический сб. М.: Иностр. лит., 1960. Вып. 1. С. 109–148.

[2] *Yubin Chen*, *Jiandong Li*, *Jiamo Chen*. A new algorithm for network probabilistic connectivity. Military Communications Conference (MILCOM 1999). Proceedings. IEEE 1999. Vol. 2. P. 920–923.

[3] *Толчан А. Я.* О связности сети. Проблемы передачи информации. 1964. Вып. 17. С. 3–7.

[4] *Ломоносов М. В.*, *Полесский В. П.* Верхняя граница надежности информационных сетей. Проблемы передачи информации. 1971. Т. VII, вып. 4. С. 78–81.

[5] *Shooman A. M.* Algorithms for network reliability and connection availability analysis. Electro/95 International. Professional Program Proceedings. 1995. P. 309–333.

[6] *Кауль С. Б.* Оценка вероятности связности случайного графа. Эффективность и структурная надежность информационных систем (СМ-7). Новосибирск, 1982. С. 3–6.

[7] *Кельманс А. К.* Некоторые вопросы анализа надежности сетей. Автомат. и телемех. 1965. Т. XXVI, № 3. С. 567–574.

[8] *Литвак Е. И.* О вероятности связности графа. Изв. АН СССР. Техн. кибернетика. 1975. № 5. С. 161–165.

[9] *Родионова О. К.* ППП ГРАФ/3. Связность мультиграфов с ненадежными ребрами (Атлас, процедуры). Новосибирск, 1982. 32 с. (Препринт АН СССР. Сиб. отделение. ВЦ; 356).

[10] *Родионов А. С.*, *Родионова О. К.* К вопросу практического использования формулы Мур–Шеннон для расчета вероятности связности локальных сетей. Тр. 2 Междунар. науч./практ. конф. «Информационные технологии и радиосети (ИНФОРАДИО-2000). Омск, 2000. С. 67–69.

[11] *Родионова О. К.*, *Герцева А. А.* О построении оптимально-связных графов. Мат. междунар. симп. по проблемам информатики, модульных систем и сетей (ICS-NET'2001). М., 2001. С. 200–204.

[12] *Родионов А. С.*, *Родионова О. К.* О точном вычислении вероятности связности графа. Тр. Междунар. конф. «Вычислительные технологии и математические модели в науке, технике и образовании», Алма-Ата, Казахстан, 18–20 сентября 2002 г. Алма-Ата, 2002. Т. 5. С. 140–147.

[13] *Rodionova O. K.*, *Rodionov A. S.*, *Choo H.* Network Probabilistic Connectivity: Exact Calculation with Use of Chains. ICCSA-2004, Springer LNCS, Vol. 3046. P. 315–324.

[14] *Родионова О. К.* Некоторые методы ускорения расчета надежности информационных сетей. Материалы XXX Международной конференции «Информационные технологии в науке, образовании, телекоммуникации и бизнесе», Украина, Гурзуф, 2003. С. 215–217.

[15] *Мигов Д. А.* Использование разрезов случайного графа для вычисления вероятности его связности. Тр. конф. молодых ученых ИВМиМГ СО РАН. Новосибирск, 2004. С. 134–141.

[16] *Мигов Д. А.* Вероятность связности 5-вершинного графа. МНПК «Связь-2004». Международный семинар «Вычислительные методы и решение оптимизационных задач». Новосибирск, 2004. С. 113–116.

[17] *Мурзин М. Ю.* Расчет надежности и вычисления коэффициентов полинома связности телекоммуникационных сетей с абсолютно надежными связями и ненадежными узлами. Тр. 8-й Междунар. конф. «Проблемы функционирования информационных сетей» (МНПК «Связь-2004»). Иссык-Куль, 2004. Т. 2. С. 213–216.

[18] *Родионова О. К.* Параллельная реализация метода точного рассчета вероятности связности сети. Мат. Междунар. семинара «Вычислительные методы и решение оптимизационных задач»: Тр. МНТК Связь-2004. 2004. Т. 3. С. 157–160.

Архитектура системы предотвращения вторжений

В. В. Платонов

За сравнительно небольшой промежуток времени системы обнаружения вторжений прошли значительный путь развития и в настоящее время вместе с межсетевыми экранами являются одним из важнейших элементов защиты корпоративных сетей. Но системы обнаружения вторжений имеют один общий недостаток, обусловленный самой природой таких систем, — их пассивность. Хотя основная модель таких систем включает в себя модуль реакции, но на него традиционно возлагают только функции оповещения (выдача предупреждения на консоль администратора, выдача сигнала на пейджер, посылка сообщения по электронной почте и т. п.). Особенно остро данный недостаток становится ощутимым с использованием высокоскоростных каналов связи, ответная реакция администратора при получении сигнала оповещения становится практически бесполезной. Поэтому в последние годы все большее внимание исследователей и компаний-производителей привлекает активный подход, приведший к созданию систем предупреждения вторжений (СПВ) [1]. В этом случае вносится элемент активного поведения системы защиты, то есть функции, дотоле присущие только межсетевым экранам. Несмотря на сравнительно короткий период существования (3–5 лет), в данной технологии можно выделить три основных этапа.

На первом этапе СПВ выполняли простейшую функцию закрытия соединения, которое было признано принадлежащим вторжению. Это реализовывалось посылкой пакетов разрыва соединения (для протоколов, базирующихся на TCP — посылкой пакета с установленным флагом FIN или флагом RST, или посылкой пакета ICMP Host Unreachible). Несмотря на множество недостатков, эта идея может использоваться и в современных системах для отдельных сетевых протоколов.

Вторым этапом развития технологии СПВ явилась организация внутренней связи с межсетевым экраном, когда сигнал об обнаружении вторжения от СПВ поступает на межсетевой экран. Получение

межсетевым экраном такого сигнала совместно с данными, характеризующими конкретный вид вторжения, приводит к изменению (добавлению) правил фильтрации, что, в свою очередь, приводит к прерыванию вторжения. Данный подход позволяет использовать многочисленные возможности и достоинства межсетевых экранов, но достаточно трудно реализуется в силу нескольких причин. Во-первых, не все межсетевые экраны поддерживают режим смены правил фильтрации «на лету», во-вторых, необходимы точные данные о вторжении от СПВ, чтобы выбрать необходимые, заранее заготовленные правила, в-третьих, продолжают «работать» известные приемы обхода межсетевых экранов и так далее. Кроме того, использование заранее заготовленных правил не позволяет защищаться от новых видов или модификаций известных вторжений. Поэтому естественным оказался переход к созданию полностью автономных активных систем.

На текущем этапе развития технологии СПВ используют два сетевых интерфейса, что позволяет им реализовывать функции автономного обнаружения и фильтрации части трафика, который принадлежит вторжению. Отметим, что и в данной технологии возможно построение хостовой СПВ, но она должна располагаться до сетевого стека защищаемого хоста. Поскольку решение СПВ принимает автоматически, то значительно возрастают требования к качеству определения наличия вторжений. Это требует новых подходов к разработке архитектуры СПВ и соответствующих механизмов обнаружения вторжений.

Предлагаемая архитектура СПВ имеет четырехуровневую (иерархическую) модульную структуру. Процесс анализа пакета (или собранного из фрагментов пакета) осуществляется из корня дерева иерархии. В зависимости от имеющейся информации, полученной от модуля данного уровня, а также от используемого протокола, происходит перемещение по дереву иерархии. На каждом уровне иерархии модули обнаружения используют различные методы и могут фильтровать «зловредный» трафик. В качестве первого (корневого) уровня используются модуль «нормализации» IP протокола и модуль обнаружения атак отказа в обслуживании. Процесс нормализации базируется на идее Паксона [2] и находит свое подтверждение в появлении методов «нормализации протоколов». Нормализация фрагментированных пакетов на этом уровне иерархии осуществляется только после реассемблирования пакета.

Процесс проведения нормализации применяется как к пакету, так и к соответствующему протоколу. В зависимости от уровня протокола (сетевой, транспортный, прикладной) происходит переход по уровням структуры. Таким образом, нормализации подвергаются IP дейтаграммы, TCP и UDP пакеты, ICMP сообщения и протоколы прикладного

уровня. Процесс нормализации связан с устранением всех нарушений в структуре и параметрах принятого пакета. Анализ различного вида Internet трафика показывает, что определенная часть пакетов посылается с нарушениями соответствующих RFC. Это вызывается как определенными недостатками соответствующих RFC, так и различными практическими реализациями сетевого стека. Основная же часть таких нарушений напрямую связана с попытками и собственно вторжениями. Всесторонний анализ структуры основных протоколов позволил четко сформулировать возможные нарушения и разработать методы их устранения. В оригинальной работе [1], например, указаны 24 «возможных нарушения» для IP, 38 для TCP, 2 для UDP и 9 для ICMP. Устранение подобных несоответствий, кроме обнаружения и предотвращения вторжений, не препятствует выполнению межсетевого обмена и повышает надежность выполнения приложений пользователей. Иллюстрацией применения данного подхода для предотвращения вторжений может послужить анализ червя Slammer, осуществившего многочисленные вторжения в январе 2003 года [3]. Этот червь реализовывал вторжение посылкой одного (!) пакета UDP длиной 376 байт на порт 1434, который позволял реализовать переполнение буфера на приемном хосте с Microsoft SQL Server 2000 и Microsoft Desktop Engine (MSDE) 2000. Использование червем уязвимости Microsoft ISS приводило выполнению зловредного кода, который, в свою очередь, посылал аналогичный пакет по адресам, формируемым по рекуррентной зависимости, что, в принципе, позволяло перебрать все возможное пространство сетевых адресов. Применение метода нормализации, к UDP в частности, позволило бы пресечь распространение данного червя.

Особенно трудной является нормализация протоколов прикладного уровня. Для построения СПВ основной упор сделан на нормализацию протокола HTTP, который служит для организации Web-взаимодействия и является основой многих других протоколов межсетевого взаимодействия. Функция нормализации реализуется с помощью специальных правил, которые могут дополняться. Правила включают «нормализацию» URI, разборку и сборку пакета HTTP с использованием заданного словаря, генерацию даты/времени в заданных форматах, анализ определенных областей пакета. Нормализация проводится в соответствии с RFC 850, 1123 и 2616.

Применение данного подхода позволяет реализовать функции дополнительной защиты для протокола HTTP, которую по аналогии с NAT можно условно назвать трансляцией имен файлов (HTTP File Name Translation). Например, при появлении обращений к файлам защищаемого Web-сервера можно заменять их имена по соответствующей таблице, что позволяет полностью скрыть от атакующего структуру файловой системы Web-сервера.

Принцип построения модуля обнаружения атак отказа в обслуживании базируется на подходе, применяемом в технологии TCP intersept компании Cisco [4]. Данный модуль содержит значения порогов (задаваемые пользователем) для числа пакетов данного вида, полученных за заданный промежуток времени, и для скорости поступления пакетов данного вида за заданный промежуток времени. Превышение значений этих порогов приводит к включению нескольких механизмов фильтрации пакетов. Отметим, что вопрос динамического изменения значений порогов в зависимости от входящего трафика нуждается в дополнительном исследовании. Кроме того, выбор соответствующего правила фильтрования (удаления пакетов) также требует дополнительных исследований. В настоящее время реализованы простейшие правила вида last in — first out.

Таким образом, в результате функционирования модулей первого уровня иерархии возможны четыре исхода.

- Пакет передается дальше (вторжения или предпосылки не обнаружено).
- Пакет нормализуется и передается дальше.
- Пакет нормализуется и передается дальше с указанием ранжированной предпосылки к вторжению.
- Пакет отбрасывается.

Далее пакет (в первых трех случаях) поступает на модули следующего уровня иерархии. Данные модули, кроме нормализации протоколов этого уровня, выполняют функции контроля состояния соединения (или виртуального соединения). На этом уровне контролируется, например, состояние соединения (устанавливается, установлено, закрывается). Для текущего состояния проводятся соответствующие проверки (например, контроль номера пакета в соединении).

На следующем уровне иерархии для каждого пакета, принадлежащего определенному соединению и, соответственно протоколу, реализуются процедуры нормализации и определенных проверок.

На четвертом уровне иерархии осуществляются проверки контекста (поля данных). Необходимо отметить, что для многих протоколов прикладного уровня построения «схемы состояний» является достаточно трудоемкой задачей. Проведение поиска определенного контекста осуществляется не во всех приходящих пакетах, а только в тех, которые относятся к соответствующему состоянию соединения данного протокола. Например, для поиска определенного контекста, использовавшегося для реализации уязвимостей в протоколе HTTP, нужно

искать данный контекст только при использовании команд GET и POST.

Таким образом, разработанная модульная трехуровневая архитектура СПВ позволяет организовать надежную защиту от вторжений. Применение уровневой структуры позволяет сократить число правил, применяемых для обнаружения вторжений. Модульное построение позволяет добавлять новые модули и модифицировать уже имеющиеся без коренной переделки системы. Отметим, что отдельные модули на разных уровнях иерархии используют лучшие решения, подтвержденные практикой применения межсетевых экранов.

Список литературы

[1] Beyond IDS: Essentials of Network Intrusion Prevention. Top Layer Networks, 2002.
[2] *Handley M.*, *Kreibich C.*, *Paxson V.* Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantic. Proc. USENIX Security Symposium, 2001.
[3] CERT Advisory CA-2003-04 MS-SQL Server Worm.
[4] Managing Cisco Network Security. Syngress Publishing, 2002.

К вопросу о выявлении возможных переполнений буферов посредством статического анализа исходного кода программ

Ф. М. Пучков, К. А. Шапченко

В данной работе предлагается небольшой обзор и классификация известных средств анализа исходного кода программ на предмет выявления возможных переполнений буферов. В качестве варианта построения подобных анализаторов приводится описание метода, реализованного авторами работы. Кроме того, в работе приводятся нижние оценки сложности работы большинства статических анализаторов, использующих графовое представление программы.

Введение

Проблема эффективной проверки программного кода является актуальной и связана, в первую очередь, с разработкой и использованием программного обеспечения с открытым исходным кодом. Кроме того, подобные программы часто используются как составляющие крупных программных комплексов с высокими требованиями к безопасности.

Аудит ПО может включать в себя методы и средства выявления различных типов ошибок и уязвимостей. В нашем исследовании рассматривается важное направление — анализ С-программ на наличие возможных переполнений буферов. По этому направлению разработан метод, позволяющий производить статическую проверку ПО (то есть, определение возможных уязвимостей — переполнений буферов — по исходному коду ПО).

Данной проблеме посвящено много исследований; представим некоторые из них. В отдельную группу можно выделить работы ITS4 [1], LCLint [2] и некоторые другие, посвященные «обучению» лексических и синтаксических сканеров. Принцип их работы — сравнение анализируемого кода с набором шаблонов потенциально опасных конструкций; обычно в качестве таких конструкций рассматривают вызовы некото-

рых библиотечных функций, в которых не производится (или не может быть произведена) проверка корректности аргументов, таких как функции `strcpy`, `sprintf`, `gets` из стандартной библиотеки C. Иногда шаблоны позволяют учитывать аргументы функций. При этом, различным шаблонам приписывается своя степень опасности. Однако, поскольку база шаблонов не может охватить все возможные варианты написания некорректных программ, эффективность таких методов довольно низка. Под эффективностью будем понимать пару показателей — процентное соотношение ложных тревог (ошибки первого рода) и пропущенных ошибок (ошибки второго рода) к количеству проверяемых обращений к памяти (таких как обращение к элементу массива по индексу, разыменовывание указателей, выделение — освобождение памяти).

Ко второй группе относим такие работы, как D. Wagner [3] и N. Dor [4]. В этих работах предлагается более глубокий, в отличие от методов первой группы, анализ. Одно из принципиальных отличий — введение условий корректности, например, соответствие индекса массива допустимым границам.

Существенно и то, что в этих работах проводится построение математических моделей, позволяющих доказать корректность работы проверяемых программ (корректность понимается в смысле отсутствия переполнений буферов), то есть они не допускают ошибок второго рода. В работе [3] в качестве базы используется анализ интервалов допустимых и фактических значений переменных. В работе [4] основным математическим аппаратом является исследование систем линейных неравенств. Необходимые данные для обработки этими алгоритмами накапливаются вследствие некоторой обработки исходного текста анализируемой программы. Методы второй группы также отличаются друг от друга сложностью и эффективностью анализа (например, методы могут учитывать или не учитывать управляющий граф программы, отличаться степенью обратимости преобразования исходного кода к внутреннему представлению). Метод, рассматриваемый далее, можно отнести к этой же группе. Далее опишем его подробнее и коснемся вопросов, связанных с оценкой сложности аналогичных подходов.

1. Описание метода

При создании метода авторы хотели обратить особое внимание на некоторые специфические моменты. Во-первых, главным требованием было создать метод, отвечающий основным характеристикам второй группы методов — формализовать понятие условие корректности и не допускать ошибок второго рода. Во-вторых, создаваемый инструмент

должен быть легко расширяемым с возможностью добавления новых алгоритмов исследования и согласования их с уже имеющимися. В-третьих, ставилась задача обеспечить эффективность подхода на различных по размеру программах, начиная от простых тестовых примеров и заканчивая сложными программными продуктами. Для достижения этих целей необходим более глубокий анализ, чем в работах [3] и [4], включающий в себя учет управляющего графа программы, использование зависимости от входа программы, поддержка интерактивности.

Опишем общую схему работы метода. Входными данными для алгоритма является исходный текст анализируемой программы на языке C. В качестве результата создается список подозрительных точек в исходной программе. Применение метода разделено на три этапа:

- этап I — преобразование исходного текста программы во внутреннее представление;
- этап II — создание информационной базы;
- этап III — вывод условий корректности и получение результатов.

Этап I отвечает за упрощение задачи, избавление от сложных конструкций языка C и переход к более простым инструкциям языка блок-схем, а в дальнейшем — к контекстному графу. Такое представление упрощает и унифицирует алгоритмы создания информационной базы, также во внутреннем представлении в удобном формате сохраняется вся необходимая для последующей обработки информация о логике выполнения программы.

На этапе II происходит анализ контекстного графа для получения условий информационной базы. Информационная база представляет собой набор систем ограничений (предикатов), связывающих переменные программы. Отрицание условия корректности добавляется как один из предикатов в каждую систему. При этом на этапе II необходимо заботиться о выполнении следующего свойства: если множество решений каждой системы информационной базы пусто, то условие корректности должно быть выполнено.

На этапе III применяются алгоритмы вывода условий корректности из условий информационной базы. Этапы II и III могут взаимодействовать друг с другом, изменяя параметры используемых алгоритмов для адаптации к конкретной задаче и оптимизации вывода условий корректности.

Целью I этапа является получение внутреннего представления анализируемого исходного кода — программы на упрощенном языке и набора условий корректности, связанного с этой программой. Упрощенный язык состоит из инструкций присваивания, условного/безусловного перехода и остановки. Будем его называть «языком блок-схем» из-за

возможности легко описывать на нем блок-схемы — простейшие средства описания и визуализации алгоритмов.

Для осуществления такого упрощения необходимо провести некоторые преобразования исходной программы. Они включают в себя:

- приведение всех циклов к одному виду;
- преобразование условных инструкций;
- разбиение сложных арифметических выражений на несколько простых;
- обработка структур и объединений;
- обработка вызовов функций (с помощью встраивания исходного кода или с помощью аннотирования вызовов, например, для библиотечных функций);
- непосредственно преобразование к языку блок-схем.

Каждое условие корректности представляется в виде двойного неравенства

$$\texttt{arr\_min} \leqslant \texttt{index} < \texttt{arr\_max},$$

где `arr_min` и `arr_max` соответствуют значениям минимального и максимального индекса, по которому возможен доступ к массиву `arr` в некоторой точке программы, фиксированной для этого условия корректности, а `index` — соответственно индекс, по которому происходит фактический доступ в этой же точке. Естественно, значения `arr_min` и `arr_max` необходимо корректно модифицировать при арифметике указателей или изменении размера захватываемой памяти. Для этого вводятся новые переменные (которые мы называем *ассоциированными*), отвечающие этим значениям, и производятся дополнительные преобразования исходного текста для отражения изменений этих переменных. После таких преобразований становится возможным построить необходимые условия корректности, например, анализируя обращения по индексу или разыменовывания.

Следующей стадией обработки C-программы является преобразование ее текста на языке внутреннего представления к виду так называемого *контекстного графа*. Эта стадия включает:

- построение графа всех переходов;
- избавление от массивов;
- получение *реквизитов* контекстного графа;
- синтаксический анализ предикатов и подстановок, присутствующих в программе.

Каждая вершина контекстного графа помечена объектом — предикатом, истинность которого проверяется в соответствующей точке программы. Аналогично, каждое ребро контекстного графа отмечено парой (символ «+» или «−», подстановка переменных). При этом символ отвечает истинности или ложности предиката, помечающего начало данного ребра, а подстановка определяет преобразование переменных, производимое при исполнении соответствующей серии инструкций присваивания.

Остановимся подробнее на некоторых из указанных этапов. Поскольку одной из важнейших задач нашего метода является проверка принадлежности индекса массивов допустимым границам, то помимо нахождения этих границ нужно учитывать возможность зависимости некоторых переменных от содержимого массивов (например, как в инструкции «`x = a[i]`»). Для этого есть два подхода. Первый состоит в том, что каждая подстановка такого вида заменятся на «`x = unknown`», где `unknown` указывает на неопределенность значения выражения. Следовательно, при таком подходе содержимое массивов не учитывается, а считается неопределенным. Другой подход — более сложный — искать для каждого массива «инициирующие блоки», то есть куски программы, где происходит изменение элементов массива. Тогда можно составить специальные таблицы, содержащие заведомо истинную информацию о содержимом массивов (случаи когда данные могут не соответствовать действительности мы отсекаем). Ниже приведены примеры таких таблиц.

Таблица 1 (*информация о массиве* `a`)

Значение индекса	$0 \ldots 99$	$100 \ldots +\infty$
Значение элемента массива	$0 \ldots +\infty$	$-\infty \ldots +\infty$

В данной таблице указано, что в элементы `a[0]`, ..., `a[99]` обладают общим свойством, а именно, эти значения принадлежат множеству $\{0, 1, 2, \ldots\}$. Про элементы массива, начиная с 100-й позиции, мы ничего сказать не можем. При этом элементы специфицируются по принадлежности индекса некоторому интервалу значений. Возможен и другой способ.

Таблица 2 (*информация о массиве* `b`)

Значение индекса	$0 \ldots$ `b_max` $- 1$
Значение элемента массива	0

Здесь имеется в виду, что в одной из позиций массива `b` с индексом, принадлежащим множеству {0, 1, ..., `b_max`}, встречается значение `0`. Переменная `b_max` отвечает за верхнюю допустимую границу индекса

массива b. Такой способ представления более эффективен при работе со строками (когда условием отсутствия переполнения является появление терминирующего символа '\0' раньше конца массива), но также применим и при работе с целочисленными массивами. Таблицы обоих видов можно использовать одновременно.

Реквизитами контекстного графа будем называть разделение его переменных на две группы. В первую группу относим переменные, значения которых не модифицируются в процессе выполнения программы. Кроме того, допускается, что их начальные значения также не определены. Такие переменные будем называть параметрами. Во вторую группу мы относим все остальные переменные. Следует заметить, что такое разделение становится полностью корректным лишь после стадии избавления от массивов.

Большинство используемых алгоритмов генерации условий информационной базы опираются на класс арифметических выражений — линейных комбинаций переменных реквизитов с коэффициентами-многочленами от параметров реквизитов. Отсюда следует, что смысл выделения параметров — расширение класса подходящих выражений.

Опишем кратко идеологию метода на этапе II. Пусть имеется некоторое множество $\{I_\alpha\}$ — алгоритмов, которые позволяют составить и проверить гипотезы о справедливости некоторых условий в различных вершинах контекстного графа (естественно, каждый алгоритм отвечает за свой «род» гипотез). Тогда, применяя сборщик «minimal_set_method», описанный подробно в статье авторов [5], позволяющий выборочно отсеивать условия, неприменимые для проверки условия корректности, и нужным образом модифицировать применимые, можно получить информационную базу. В рассматриваемом методе класс линейных арифметических выражений является основой всех алгоритмов множества $\{I_\alpha\}$. Это значит, что от количества выражений, представимых в таком виде, существенно зависит эффективность всего метода.

Для анализа системы ограничений предлагается применять такую последовательность действий. Сначала проводится сопоставление всевозможных пар предикатов на случай выявления элементарных противоречий и отсечения лишних условий. Предикаты при этом могут иметь произвольный вид. Затем, если этот тест не дал отрицательного ответа на вопрос о существовании решения, то из системы выбрасываются все предикаты, не представимые в виде линейных параметризованных уравнений или неравенств (с коэффициентами-многочленами от параметров реквизитов). Для исследования редуцированной системы можно использовать «параметрический» вариант симплекс-метода. Одним из

препятствующих обстоятельств является отсутствие полной упорядоченности кольца многочленов нескольких переменных. Следовательно, в таком виде симплекс-метод не всегда применим.

Поэтому, в случае когда этот тест не дал удовлетворительного результата, возможно применение вероятностных соображений. Предлагается такой подход: задается семейство распределений (для каждого параметра — свое) и количество тестируемых выборок. После подстановки в систему значений параметров из некоторой выборки получим уже обыкновенную (непараметризованную) систему линейных уравнений и неравенств. Каждая такая система решается классическим симплекс-методом. Отклонение от принятой концепции — недопущение ошибок второго рода — при вероятностном подходе минимально: варьируя количество выборок, всегда можно добиться ситуации, в которой вероятность такой ошибки будет меньше любого наперед заданного $\varepsilon > 0$.

Считаем, что исходная параметрическая система неразрешима, если для всех выборок полученные непараметрические системы не имеют решений. Количество выборок и сами вероятностные распределения задает пользователь в интерактивном режиме.

2. Оценки сложности алгоритмов верификации

С практической точки зрения интересен вопрос оценки времени работы статических методов анализа, аналогичных предложенному выше. В этом разделе докажем несколько фактов, касающихся оценок сложности методов, основанных на анализе управляющего графа программы.

При анализе управляющего графа программы возникает важная задача — поиск возможных путей из одной вершины графа в другую (например, для построения возможных «трасс» выполнения). Мы рассмотрим задачу оценивания сверху количества *простых* путей в управляющем графе программы.

Вначале приведем несколько определений.

Определение 1. *Управляющим графом* программы назовем конечный ориентированный граф G (возможно содержащий кратные ребра, а также ребра-петли), в котором

- V — множество вершин, $|V| \geqslant 2$. В графе выделены две особые вершины F_0, F_1 (вход и выход из программы). Множество, включающее эти две вершины обозначим через F.
- E — множество ребер. Каждая вершина $V \setminus F$ имеет исходящую степень 2. Вершина F_0 имеет исходящую степень 1, входящую степень 0, вершина F_1 имеет исходящую степень 0.

Замечание 7. Вершины графа, таким образом, отвечают условным переходам в программе: проверкам условий (`if`) и циклам (`for`, `do`, `while`).

Определение 2. Подмножество M множества вершин V управляющего графа G называется *подчиненным* вершине $A \in M$, если выполнены два свойства.

1. Ограничение G на множество вершин M является сильно связным графом, содержащим хотя бы одно ребро.
2. В графе нет ребер вида (B,C) или (C,B), где $B \in V \setminus M$, $C \in M \setminus A$.

Если это так, то вершина A называется *начальником* каждой вершины $X \in M$.

Замечание 8. Отметим, что $M = \{A\}$ тогда и только тогда, когда в графе существует ребро вида (A, A). Кроме того, для вершины $X \in M$, $X \neq A$ исходящие ребра могут вести только в вершины M. Аналогично, для вершины $Y \in V \setminus M$ ребра могут вести только в $\{A\} \cup (V \setminus M)$.

Определение 3. Вершина A управляющего графа называется *функциональной*, если для нее существует непустое подчиненное множество.

Замечание 9. Будем рассматривать только такие управляющие графы в которых из вершины F_0 достижимы все другие вершины, а также, из любой вершины достижима вершина F_1. Это не является существенным ограничением, поскольку, например, можно исключить из рассмотрения недостижимые точки программы.

С учетом сделанного замечания докажем несколько свойств функциональных вершин.

Утверждение. *Пусть A — функциональна, M — ее подчиненное множество, B, C — соседи A (то есть в графе существуют ребра видов (A, B), (A, C)). Тогда*

1. *Ровно одна из вершин B, C лежит в M.*
2. *Если два пути L_1, L_2, выходящих из вершин B и C соответственно, не проходят через вершину A, то они не пересекаются.*
3. *Отношение подчиненности является частичным порядком на множестве $V \setminus F$.*

Доказательство. 1. Предположим противное. Пусть сначала $B, C \notin M$. Тогда найдется $X \in M \setminus A$, поскольку ребер вида (A, A) в графе нет. Значит, ограничение G на M не является сильно связным графом, так как содержит изолированную вершину A. Противоречие. Если $B, C \in M$, то легко понять, что нарушено требование существования пути из вершины A в вершину F_1: из каждой вершины множества M ребра ведут только в M.

2. Воспользуемся только что доказанным свойством. Без ограничения общности будем считать, что $B \in M$, $C \notin M$. Доказательство проведем индукцией по сумме длин путей L_1 и L_2. База очевидна. Пусть B', C' — следующие за B, C вершины в L_1, L_2 соответственно. Тогда, поскольку B, C, B', $C' \neq A$, то $B' \in M$, $C' \notin M$, и мы пришли к той же ситуации, но сумма длин путей уменьшилась. При этом мы пользовались лишь тем, что B, C отличны от A и ровно одна из этих вершин лежит в M, что верно и для вершин B' и C'. Таким образом, свойство 2 доказано.

3. Достаточно доказать свойства антисимметричности и транзитивности (рефлексивность — очевидна). Предположим, что A — начальник B с подчиненным множеством M_A, B — начальник A с подчиненным множеством M_B ($A \neq B$). Тогда $A, B \in M_A \cap M_B$. По определению подчиненного множества оба исходящих ребра вершины A ведут в M_B. Аналогично, оба исходящих ребра вершины B ведут в M_A, а для остальных вершин множества $M_A \cup M_B$ исходящие ребра ведут в это же множество. Поскольку $F_1 \notin M_A \cup M_B$, получаем противоречие с условием существования пути из A в F_1. Пусть теперь A — начальник B, B — начальник C. Случаи $A = B$ и $B = C$ очевидны, поэтому считаем, что $A \neq B$ и $B \neq C$. M_A, M_B, M_C — подчиненные множества вершин A, B, C соответственно. Заметим, что $A \neq C$ по доказанному свойству антисимметричности. Достаточно доказать, что $C \in M_A$. В силу свойства 1 определения 2 имеем: существует путь L, соединяющий B с C, лежащий в M_B, следовательно не проходящий через вершину A. Предположим, что L не лежит в $M_A \setminus A$, и пусть X — первая вершина: $X \notin M_A \setminus A$, $X \neq B$, Y — предшествующая ей вершина. Тогда $Y \in M_A \setminus A$. По свойству 2 определения 2 имеем $X \in M_A$. Следовательно, $X = A$. Противоречие. Значит, L и, в частности, вершина C лежат в M_A.

Утверждение полностью доказано. □

2.1. Лемма об оценке числа простых путей в управляющем графе

Пусть, как и раньше, G — управляющий граф. Введем обозначение: $FV \subset V$ — множество функциональных вершин графа, $|V \setminus (FV \cup F)| = n$ — количество нефункциональных вершин графа, отличных от F_0, F_1. $L_G(A, B)$ — множество простых путей в графе G, соединяющих A и B.

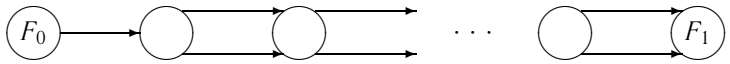
Лемма. *Для произвольных вершин A, B справедлива оценка*

$$|L_G(A, B)| \leqslant 2^n. \tag{1}$$

Доказательство. Покажем, что выбор вершины B позволяет однозначно «ориентировать» каждую функциональную вершину в том смысле, что у каждой функциональной вершины можно отбросить одно исходящее ребро (получив новый граф H) так, что для любой вершины $A \in V$ получим $L_G(A, B) = L_H(A, B)$. Для графа H оценка (1) очевидна, поскольку количество простых путей в графе между двумя вершинами не превосходит произведения всех положительных исходящих степеней вершин графа.

Пусть X — функциональная вершина, Y, Z — ее соседи. Если $X = Y$ или $X = Z$, то отбросим ребро вида (X, X). Очевидно, что такое ребро не может входить ни в один простой путь из A в B. Пусть $X \neq Y$, $X \neq Z$ и пусть существует простой путь L из X в B. Без ограничения общности считаем, что он проходит по ребру (X, Y). Тогда докажем, что не существует простого пути из X в B, проходящего по ребру (X, Z). Предположим противное: L' — такой путь. Отбросим в путях L, L' начальные ребра. Останутся простые пути L_1, L'_1, ведущие в вершину B. Заметим, что L_1, L'_1 уже не проходят через вершину A (в силу простоты путей L, L'). Следовательно, согласно пункту2 утверждения, эти пути не пересекаются. Противоречие. Удалим для вершины X исходящее ребро (X, Z). По только что доказанному свойству такая операция не влияет на множество простых путей из X в B и, следовательно, из A в B. Таким образом, возможность построения графа H доказана, что завершает доказательство леммы. □

Замечание 10. Оценка (1) — точная. Для произвольного n рассмотрим граф, состоящий из $n+2$ вершин как показано на рисунке.



Ясно, что все вершины, кроме крайних, нефункциональны, и количество простых путей из F_0 в F_1 равно 2^n.

2.2. Оценка количества нефункциональных вершин управляющего графа

Доказанная в предыдущем разделе оценка количества простых путей в управляющем графе тем не менее не дает представления о целесообразности применения статических анализаторов, основанных на исследовании управляющего графа. Для такого представления необходимо получить оценки количества нефункциональных вершин контекстного графа через количество операторов языка C, порождающих условные/безусловные переходы. Мы рассмотрим использование в языке операторов `for`, `while`, `do`, `if`, `continue`, `break`, `goto`.

Во-первых, отметим, что присутствие операторов `goto` может непредсказуемо изменять количество нефункциональных вершин управляющего графа. Для примера рассмотрим следующий фрагмент кода.

```
......
l1:
for(;; ++i)
  for(;; ++j)
    for(;; ++k)
      if (i+j+k % 2)
         goto l1;
      else
         goto l2;
l2:
......
```

Несмотря на тот факт, что оператор `for` сам по себе порождает функциональную вершину, присутствие оператора `goto` делает все три вершины управляющего графа, порожденные `for`-циклами, нефункциональными. Поэтому, далее будем предполагать, что операторов `goto` в программе нет.

Во-вторых, конструкции `for`-, `while`-циклов эквивалентны с точки зрения управляющего графа. Легко убедиться, что в случае, когда тело цикла такого вида не содержит операторов `break`, вершина управляющего графа, соответствующая проверке условия выхода из цикла, будет функциональной. При применении `do`-циклов существуют два способа построения управляющего графа.

- цикл
  `do { <тело цикла> } while (<условие>);`
  сначала преобразуется к `while`-циклу путем вынесения первой итерации выполнения тела цикла за цикл. При этом, вершина, соответствующая условию выхода из цикла будет функциональной в тех же предположениях (тело цикла не содержит операторов `break`).
- цикл моделируется естественным образом: проверка выхода осуществляется в конце тела цикла. При таком подходе вершина, соответствующая условию выхода будет нефункциональной.

Рассмотрим операторы `if`, `if-else`. С точки зрения управляющего графа такие конструкции эквивалентны. При этом, ясно, что вершина, отвечающая проверке условия не является функциональной, поскольк-

ку оба логических пути при выходе из тела цикла «сливаются» друг с другом. Следовательно, вершина не может быть функциональной в тех же предположениях (по пункту 2 утверждения), что тело цикла не содержит операторов `goto`, `break`, `continue`.

Теперь рассмотрим случаи присутствия операторов `continue`, `break`. Поскольку оператор `continue` не выводит за пределы самого внутреннего цикла, содержащего его, то этот оператор не влияет на количество нефункциональных вершин. В отличие от `continue`, оператор `break` завершает самый внутренний цикл, содержащий его, но не выводит за пределы других циклов. Следовательно, наличие этого оператора увеличивает количество нефункциональных вершин не более, чем на 1.

Таким образом, можно сформулировать следующую теорему.

Теорема. *Предположим программа содержит из всех операторов условного или безусловного перехода только операторы* `if`, `for`, `while`, `break`, `continue` *в количестве* N_1, N_2, N_3, N_4, N_5 *соответственно. Тогда для количества нефункциональных вершин* n *в управляющем графе такой программы справедлива оценка*

$$n \leqslant N_1 + N_4,$$

не зависящая от N_2, N_3, N_5, *и, следовательно, сложность анализа всевозможных трасс, согласно лемме, не превышает* $2^{N_1+N_4}$.

Данная теорема позволяет очень быстро оценить целесообразность рассмотрения всего множества возможных трасс выполнения. Как она показывает, сложность такого анализа может расти экспоненциально. Поэтому при статическом анализе необходимо использовать различные эвристические методы ограничения перебора. Одно из полезных наблюдений состоит в том, что поведение программы в некоторой фиксированной точке часто зависит не от всей пройденной с начала выполнения трассы, а лишь от ее отрезка, лежащего в некоторой небольшой окрестности данной точки.

Заключение

В заключении отметим, что параллельно с развитием теоретических концепций изложенного метода создается практическая реализация. Упрощенная версия ее была создана в начальной стадии исследований и показала хорошие результаты на нескольких базовых примерах. В настоящее время на основе существующего опыта, с учетом новых идей реализуется полная версия, предназначенная для обработки и аудита кода, в том числе и больших программных комплексов.

Список литературы

[1] *Viega J., Bloch J. T., Kohno T., McGraw G.* ITS4: A Static Vulnerability Scanner for C and C++ Code. Proceedings of the 16th Annual Computer Security Applications Conference, 2000, p. 257.

[2] *Larochelle D., Evans D.* Statically Detecting Likely Buffer Overflow Vulnerabilities. Proceedings of the 10th USENIX Security Symposium, 2001, pp. 177–190.

[3] *Wagner D., Foster J., Brewer E., Aiken A.* A First Step Towards Automated Detection of Buffer Overrun Vulnerabilities. Proceedings of the 2000 Network and Distributed Systems Security Conference, February 2000, pp. 3–17.

[4] *Dor N., Rodeh M., Sagiv M.* Cleanness Checking of String Manipulations in C Programs via Integer Analysis. In the 8th International Symposium on Static Analysis, July 2001, pp. 194–212.

[5] *Пучков Ф. М., Шапченко К. А.* Статический метод анализа программного обеспечения на наличие угроз переполнения буферов. Готовится к публикации.

Операционный анализ безопасности поведения программного обеспечения

С. С. Корт

Для того чтобы рассмотреть основы операционного анализа поведения программного обеспечения (ПО) проанализируем существующие механизмы, ограничивающие поведение программ. К таким механизмам можно отнести ограничения на поведение ПО, накладываемые со стороны операционной системы, и ограничения, накладываемые на приложения специализированными средами (типа Java).

В современных защищенных операционных системах уровень представления субъектов и объектов в операциях ограничивается ОС. Данный механизм называется монитором безопасности пересылок (Security Reference Monitor). Основной характеристикой монитора безопасности пересылок является то, что он или разрешает запрос на доступ, или запрещает его, возможно, уведомляя об этом субъекта. Монитор безопасности пересылок может быть описан в терминах функции с запросами на обслуживание, разрешениями доступа, другими компонентами состояния системы на входе (т. е. элементами области определения) и разрешениями или запретами на обслуживание на выходе (т. е. элементами области значения). Таким образом, программное обеспечение может обращаться к различным объектам системы только с использованием строго определенного интерфейса, предоставляемого ОС (интерфейс прикладного программирования).

Поведение программ ограничивается не только правилами доступа к объектам системы, но и множеством ограничений, накладываемых политикой безопасности системы и реализуемой монитором безопасности пересылок.

Таким образом, формируется множество поведений программы, разрешенных со стороны ОС. Необходимо отметить, что монитор безопасности пересылок устанавливает «глобальную» политику безопасности в системе, т. е. политику безопасности, учитывающую субъекты в виде учетных записей пользователя.

К достоинствам данного подхода можно отнести следующие положения:

- независимость безопасности поведения программы от языка программирования, на котором написана программа;
- независимость безопасности поведения программы от внешних механизмов;
- высокая производительность.

К недостаткам данного подхода можно отнести следующие моменты:

- правила политики безопасности вычислительной системы ограничивают поведение пользователей, а не программ, вследствие чего теряется гибкость;
- проблемы безопасности, связанные с поведением неизвестного программного обеспечения;
- ограниченный набор политик безопасности, поддерживаемый монитором безопасности пересылок.

Альтернативой подходу, основанному на применении монитора безопасности пересылок, является подход организации «песочницы», примененный в языке Java. Данный подход применяется к приложениям Java-аплетам. Данные приложения может использовать любой клиент, который вовсе не обязан знать правила «техники безопасности» при работе с этими небольшими программками. Именно поэтому для аплетов предусмотрены самые жесткие методы защиты. В данном подходе удачно определены подсистемы, доступ к которым невозможен для неизвестного ПО. Кроме того, необходимо отметить, что описанные выше ограничения могут быть ослаблены и установлены для отдельных программ.

К преимуществам данного подхода можно отнести следующие положения:

- создание доверенной среды для выполнения приложений пользователя;
- возможность выполнения программ, о поведении которых ничего не известно.

К недостаткам данного подхода можно отнести следующие моменты:

- зависимость безопасности поведения программ от языка программирования, на котором написана программа;
- жесткие ограничения на поведение программ вследствие отсутствия учета спецификаций программы.

Предлагаемый в данной работе подход позволяет, наследуя достоинства подхода, описанного при рассмотрении монитора безопасности пересылок и подхода, разработанного для мобильных агентов, организовать политику безопасности, применимую к отдельным приложениям. Данный подход дополняет концепцию монитора безопасности пересылок (определяющего «глобальную» политику безопасности, примени-

мую к пользователям), позволяя сформировать политику безопасности «локальную» для приложений, и может быть использован совместно с ним.

При использовании данного подхода можно организовать «песочницу» аналогичную песочнице, используемой в подходе обеспечения безопасности программ мобильных агентов.

Ожидаемое поведение программы может быть выражено как множество функций, выполняемых программой в соответствии со спецификацией на программу. Вместе с тем ожидаемое поведение ПО может отличаться от истинного поведения. И истинное поведение ПО может при этом быть небезопасным.

Поведение программы, разрешенное ОС, ограничивает способы доступа программ пользователя к объектам системы. В свою очередь, реализация политики безопасности в вычислительной системе отделяет множество безопасных операций программы (не нарушающих политику безопасности) от множества небезопасных операций. Ожидаемое поведение программы (определенное в соответствии со спецификациями на нее) может отличаться от истинного поведения программы. И истинное поведение программы может быть при этом небезопасным.

Таким образом, для того, чтобы выполнить операционный анализ безопасности программ необходимо описать в терминах операций ожидаемое поведение программы. Тогда отклонение истинного поведения программы от ожидаемого поведения повлечет нарушение локальной или глобальной политики безопасности.

В качестве субъекта в операционном анализе безопасности поведения программ выступает программа. Программа выполняет операции над объектами системы. В качестве объектов рассматриваются ресурсы, с которыми оперирует программа.

Для того чтобы описать поведение программы можно использовать понятие «профиль». Под *профилем* понимается множество операций, выполняемых программой. Операционный анализ безопасности поведения ПО базируется на понятиях эталонного и операционного профиля. При этом *эталонный профиль* — профиль, описывающий множество разрешенных для программы поведений в терминах операций программы над ресурсами. В свою очередь *операционный профиль* является профилем, получаемым в результате записи операций, выполненных программой во время исполнения.

Таким образом, в результате сравнения эталонного и операционного профиля можно оценить безопасность поведения программы как отклонение операционного профиля программы от эталонного профиля. Относящиеся к безопасности аспекты поведения программы можно

удобно описать с использованием логики предикатов — составить спецификацию программы.

Формат профиля определяется множеством операций, определенных для программы. К операциям можно отнести: чтение, запись, выполнение, удаление, копирование и т. д.

Чтобы иметь возможность описать профиль программы с использованием описанных выше операций необходимо выполнить анализ и группирование вызовов прикладного интерфейса программирования.

При анализе безопасности были выделены следующие группы интерфейсов, оказывающие влияние на безопасность системы: работа с окружением и реестром, работа с файловой системой, управление процессами, работа с сетью, разное.

В результате каждая функция прикладного интерфейса программирования с помощью которой можно осуществить операцию, оказывающую влияние на безопасность системы должна быть отнесена к одному из типов операций, составляющих профиль.

Множество операций, а также группирование операций функций прикладного интерфейса предопределено, но может редактироваться администратором системы.

В эталонном профиле описывается множество операций и ограничений на них. Данные ограничения описываются с использованием различных политик безопасности программы. При этом политики безопасности, описанные в профиле программы, будут «локальными» для программы.

К «локальным» политикам безопасности могут быть предъявлены следующие требования:

- непротиворечивость по отношению к глобальной политике безопасности; определение политики безопасности, противоречащей политике безопасности, определенной с использованием монитора безопасности пересылок, нецелесообразно, т. к. приводит к поведению программы, нарушающему безопасность системы;
- реализуемость — не всякая политика безопасности может быть определена при описании эталонного профиля;
- определенные в эталонном профиле политики безопасности должны быть сформулированы с использованием принципа наименьших привилегий.

Для составления эталонного профиля поведения программы возможно два подхода:

1. Администратор системы определяет правила, соответствующие выбранным им политикам безопасности, для определенных операций. Данные правила должны формулироваться с учетом ожи-

даемого поведения программы, определяемого на основе спецификаций на программу.

2. Альтернативным способом настройки системы является использование механизма обучения, создающего предикатные выражения во время тестовых запусков программы. Таким образом, эталонный профиль создается на основе операционного профиля. Администратор системы при этом должен, возможно, на основе просмотра данных о функционировании программы, признать запуск «безопасным».

Механизм, заложенный в основу операционного анализа должен просто проверять соответствие операционного профиля эталонному профилю. В случае несоответствия операционного профиля эталонному профилю считается, что программа не безопасна.

В соответствии с требованиями документа «Защита от несанкционированного доступа к информации. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей» динамический анализ исходных текстов программ должен включать следующие технологические операции:

- контроль выполнения функциональных объектов (*процедур*, *функций*);
- сопоставление фактических маршрутов выполнения функциональных объектов (*процедур*, *функций*) и маршрутов, построенных в процессе проведения статического анализа.

Предлагаемый операционный анализ безопасности поведения программного обеспечения позволяет решить данные задачи.

В СПбГПУ реализована система «Мелкоскоп», в основу которой положен описываемый в настаящей работе метод. Система «Мелкоскоп» разработана с использованием технологии клиент-сервер и предназначена для анализа безопасности поведения прикладного программного обеспечения, функционирующего под управлением ОС Windows 2000 и Windows XP.

SOFTLINE-M LTD.

ЗАО «Софтлайн-М» Адрес: 117192 г.Москва, Мичуринский проспект, 1.
тел./факс (095) 932-8958; 939-2096.
E-mail:yava@softline-m.com

ЗАО «Софтлайн-М» — системный интегратор, объединяющий и представляющий интересы различных российских компаний и компаний стран — членов СНГ.

Основные направления деятельности:

- Построение защищенных мультисервисных сетей связи, call-центров, систем IP-телефонии и видеоконференцсвязи;
- Разработка заказного ПО, создание комплексных автоматизированных информационных систем;
- Управленческое и ИТ-консультирование;
- Комплексная защита информации;
- Сервисное обслуживание, техническое сопровождение информационных систем, ИТ-аутсорсинг;
- Предоставление услуг по сборке печатных плат методом поверхностного монтажа (SMT) на новейшем оборудовании.

ЗАО «Софтлайн-М» предлагает широкий спектр услуг и решений для крупных и средних предприятий различных секторов экономики.

Офисы компании находятся в Москве, Минске, Астане и в свободной экономической зоне «Брест» (Беларусь). Производственные мощности сконцентрированы СЭЗ «Брест». СЭЗ позволяет получить существенные льготы и привилегии при производстве заказного ПО, а так же при производстве изделий и оборудования. Предлагаем услуги по контрактному производству изделий.

Компания выпускает различные изделия собственной разработки. Одна из разработок — первичный мультиплексор. Еще одной разработкой является система шифрования IP-пакетов.

Система шифрования IP пакетов предназначена для обеспечения безопасности в рамках корпоративных сетей, а также для организации виртуальных частных сетей. Система базируется на международных

стандартах IP-безопасности. Система поддерживает протокол IPsec. Реализует криптографические функции:

1. **шифрование:** ГОСТ 28147-89 (стандарт СНГ), 3DES, (возможна реализация любого необходимого потребителю алгоритма);
2. **хэширование:** ГОСТ Р34.11 (Стандарт РФ), SHA-1 (SHA-256, SHA-384, SHA-512) (Стандарт США), СТБ 1176.1-99 (Стандарт РБ);
3. **цифровая подпись:** ГОСТ Р34.10-2001 (Стандарт РФ), ГОСТ 34.310-95 (Стандарт Украины), СТБ 1176.2-99 (Стандарт РБ);
4. **аппаратный двухканальный генератор случайных последовательностей**;
5. **система создания общих секретных ключей на основе открытого распределения ключей (протокол Диффи-Хелмана).**

Ассоциация Защиты Информации

Межрегиональная общественная организация «Ассоциация защиты информации» (АЗИ) образована в 2002 году по инициативе ФАПСИ и Гостехкомиссии России. Деятельность АЗИ направлена на создание благоприятных условий для реализации потребностей граждан, бизнеса и органов государственной власти в продуктах и технологиях защиты информации.

АЗИ активно взаимодействует с аппаратом Совета Безопасности РФ, ФСБ России, Федеральной службой технического и экспортного контроля (ФСТЭК), Федеральным агентством по информационным технологиям (ФАИТ), другими министерствами и ведомствами, а также со многими финансово-экономическими структурами.

Устав АЗИ дает право осуществлять международные связи, разрешает вступать в международные общественные объединения, а также осуществлять внешнеэкономическую деятельность.

АЗИ является лицензиатом ФСБ России и Гостехкомиссии России, что дает ей право:

- осуществлять мероприятия и оказывать услуги по технической защите конфиденциальной информации;
- вести деятельность по разработке и (или) производству средств защиты конфиденциальной информации;
- осуществлять работы, связанные с использованием сведений, составляющих государственную тайну;
- вести разработку, производство, осуществлять техническое обслуживание и распространение шифровальных (криптографических) средств информационных и телекоммуникационных систем.

Ассоциация готова оказывать содействие в налаживании деловых контактов и связей с целью реализации продуктов и технологий защиты информации, обмена деловой информацией, осуществления совместных разработок и промышленного производства, проведения симпозиумов, конференций, выставок, семинаров, организации обучения специалистов в области информационной безопасности.

Предприятия, представленные в Ассоциации, предоставляют полный комплекс услуг по созданию и сопровождению интегрированных комплексных систем безопасности. Работы выполняются как с предпро-

ектного этапа, так и на любом этапе создания и реконструкции объектов. При выполнении работ обеспечивается защита категорированных и обычных объектов конфиденциальной информации и информации, содержащей сведения, составляющие государственную тайну.

Предприятия, представленные в «Ассоциации защиты информации»:

- Ассоциация ЕВРААС
- Журнал «ИНТЕГРАЛ»
- ЗАО «Ай Ди эС - Технолоджи»
- ЗАО «Аладдин Р.Д.»
- ЗАО «Диалог-Наука»
- ЗАО «Инфосистемы Джет»
- ЗАО «ИНЭЛТ»
- ЗАО «Комфакс»
- ЗАО «МВП “СВЕМЕЛ”»
- ЗАО «МО ПНИЭИ»
- ЗАО «РНТ»
- ЗАО «Санкт-Петербургский РЦЗИ»
- ЗАО «Фирма НТЦ КАМИ»
- ЗАО «СекьюрИТ»
- ЗАО «ЭВРИКА»
- НПП «СТЗИ»
- ОАО «Андэк»
- ОАО «Инфотекс»
- ОАО «ОПТИМА»
- ОАО «ЭЛВИС ПЛЮС»
- ОКБ САПР
- Компания «СИСТЕМАТИКА»
- ООО «Валидата»
- ООО «Группа Компаний СПУРТ ЛТД»
- ООО «ИКБ»
- ООО «ИнфоКрипт»
- ООО «Крипто Про»
- ООО «ЛИССИ»
- ООО «МК-ЦБ»
- ООО «СТЭЛ-Компьютерные системы»
- ООО «Центр электронных услуг», г. Уфа
- ООО Фирма «АНКАД»
- ФГУП «Калуга-прибор»
- ФГУП «Концерн “СИСТЕМПРОМ”»
- ФГУП «НИИ “Автоматики”»
- ФГУП «НИИ “Квант”»
- ФГУП «НПП “ГАММА”»
- ФГУП «Пензенский филиал НТЦ “Атлас”»
- ФГУП «ПНИЭИ»
- ФГУП «Краснодарский филиал НТЦ “Атлас”»
- ФГУП «Нижегородский филиал НТЦ “Атлас”»
- ФГУП «НТЦ “Атлас”, филиал в г. Санкт-Петербурге и Ленинградской области»
- ФГУП «ЦНИИчермет им. И. П. Бардина»
- «Удостоверяющий центр», г. Санкт-Петербург

Адрес: 125438, Москва, 4-й Лихачевский пер., д. 15,
ФГУП «НИИ “Квант”», МОО АЗИ
Телефон/факс: +7 (095) 156-7102
Телефон: +7 (095) 154-6155, 782-3357,
http://www.azi.ru,e-mail:azi@azi.ru

ДиалогНаука

ЗАО «ДиалогНаука»

ЗАО «ДиалогНаука» более двенадцати лет занимается разработкой средств борьбы с компьютерными вирусами. Высокое качество предлагаемых нами продуктов оценили миллионы индивидуальных и тысячи корпоративных клиентов в России, странах ближнего и дальнего зарубежья. Сегодня ДиалогНаука предлагает антивирусные программы семейства Doctor Web.

Антивирусные программы семейства Doctor Web, построенные на основе передовых антивирусных технологий, обеспечивают надежную защиту от компьютерных вирусов всех типов. Важным свойством программ семейства Doctor Web является модульный принцип построения, что обеспечивает одинаково надежную защиту для пользователей различных операционных систем.

Зависимыми от конкретных систем являются лишь оболочки. Ядро может быть подключено к различным оболочкам и приложениям, что обеспечивает легкость интеграции антивирусной проверки с пользовательскими прикладными задачами.

Программы Doctor Web регулярно показывают отличные результаты в авторитетных международных тестах. Так, за последние три года Doctor Web девять раз «антивирусного мира», присуждаемой редакцией ведущего международного журнала «Virus Bulletin». В тестах Virus Bulletin в феврале 2002, а также в тестах Virus Test Center при Гамбургском университете в марте 2002, в которых участвовали лучшие антивирусы со всего мира, Doctor Web был отмечен как показавший наилучшие результаты.

Существенными особенностями программ Doctor Web являются оригинальный эвристический анализатор и эмулятор процессора, позволяющий обнаруживать сложные шифрованные и полиморфные вирусы, которые невозможно найти простым поиском сигнатур. Эвристический анализатор Doctor Web выявляет вирусы, которые не описаны в используемой версии вирусной базы (впрочем, вирусные базы обновляются очень часто путем выпуска так называемых «горячих дополнений»). Кроме того, программа Doctor Web обнаруживает вирусы внутри архивов, упакованных и вакцинированных файлов, в файлах документов для MS WinWord и Excel, а также в файлах, полученных по электронной почте, из Internet, по локальной сети и из других источников.

Встраивание антивирусной проверки в разные системы — один из путей повышения уровня «вооруженности» в борьбе с вирусами и другими вредоносными программами. Вот примеры встраивания антивируса Doctor Web в прикладные системы. В системе документооборота «Сапиенс» фирмы «НПО Машиностроения» ядро Doctor Web проверяет каждый файл перед его загрузкой на хранение в базу данных. В программе Nero немецкой фирмы Ahead ядро Doctor Web проверяет каждый файл перед его «прожиганием» на CD-R. В программе Stop-Sign американской фирмы Acceleration ядро Doctor Web используется для on-line антивирусной проверки компьютеров. В почтовых системах российских фирм Mail.ru, Яндекс, Росбизнесконсалтинг, Зенон, Фактор и многих других антивирусные фильтры программы Doctor Web выполняют проверку почтовых сообщений. В китайском антивирусе iDuba.net используются два ядра — российское Doctor Web иль китайское. С 2002 года в Южной Корее и Японии продается корейский антивирус Virus Chaser, построенный на базе ядра Doctor Web.

ДиалогНаука пятый год подряд предоставляет бесплатную лицензию на использование некоторых версий антивирусных программ Doctor Web для всех государственных школ, университетов и других учебных заведений, входящих в систему Минобразования РФ. Сеть компьютеров в рамках министерства образования, которая достаточно полно покрывает территорию страны, оказывает помощь по выявлению новых вирусов и тем самым помогает предотвратить разрастание новых вирусных эпидемий.

Программные продукты Doctor Web используют такие общенациональные структуры как Администрация Президента РФ, Аппараты Правительства, Совета Федерации и Государственной Думы, ГАС «Выборы», ГУИР ФАПСИ, Центробанк и Сбербанк, Министерства обороны, образования, экономразвития, финансов, промышленности и др.

ЗАО «ДиалогНаука» имеет лицензии Гостехкомиссии на деятельность в области защиты информации и лицензии ФАПСИ на право осуществлять проектирование и производство средств защиты информации, предназначенных для использования в органах государственной власти, а также соответствующие сертификаты соответствия на антивирусные программы семейства Doctor Web от Гостехкомиссии, ФАПСИ и Минобороны.

ЗАО «ДиалогНаука»
Адрес: 119991 Москва, ул. Вавилова 40, ВЦ РАН, офис 103
Тел.: (+7-095) 135-6253, 137-0150, Тел./факс: (+7-095) 938-2970
E-mail: antivir@DialogNauka.ru
WWW-сервер: http://www.DialogNauka.ru, **FidoNet: 2:5020/69**

ФГУП «Пензенский филиал НТЦ “Атлас”»

Пензенский филиал ФГУП «Научно технический центр “Атлас”» (ПФ НТЦ «Атлас») имеет более чем 40 летний опыт работы.

Направление деятельности комплексное решение проблем обеспечения информационной безопасности, в том числе с использованием криптографии.

Для выполнения всего комплекса задач ПФ НТЦ «Атлас» обладает необходимыми лицензиями и аккредитациями ФСБ, ФАПСИ, Гостехкомиссии России, Министерства обороны РФ.Предприятие имеет высококвалифицированные кадры, в том числе с учеными степенями и необходимое высокотехнологичное оборудование для:

- проведения комплекса работ по тематическому сопровождению НИОКР в части обеспечения требуемых криптографических, инженерно-криптографических, и специальных качеств разрабатываемых изделий на всех этапах разработки;
- проведения сертификационных испытаний в «Системе сертификации средств криптографической защиты информации» РОСС RU 0001.030001 в качестве специализированной организации 1-й категории;
- проведение исследований программного обеспечения на соответствие реальных и декларируемых возможностей, соответствие профилю безопасности;
- анализ встроенного (BIOS) и системного программного обеспечения для удовлетворения требованиям безопасности обрабатываемой информации;
- проведения работ по аттестации комплексов шифрованной связи, включая работы по оценке влияния оборудования шифрованной связи на шифртехнику (в том числе на объектах Заказчика);
- проведения работ по аттестации объектов информатизации по требованиям безопасности информации;
- проведения спецпроверки технических устройств на наличие закладочных устройств съема и передачи информации;
- проведения спецобследований помещений;
- разработки шифртехники;
- разработки аппаратуры передачи данных;
- обеспечения информационной безопасности корпоративных информационных систем;
- разработки политики информационной безопасности;

Основными заказчиками работ являются ФАПСИ, МО РФ, МПС, МВД, ФСБ, Минатом. У филиала сложились устойчивые связи с научными организациями и предприятиями, разрабатывающими и изготавливающими спецтехнику, системы и комплексы обработки и передачи информации.

Работы проводятся в строгом соответствии с законодательством Российской Федерации, действующими нормативно-методическими документами уполномоченных федеральных органов в области защиты информации.

440601, г. Пенза, ул. Советская, дом 9,
Тел.: (841-2) 56-39-16, Факс: (841-2) 56-23-71,
E-mail: atlas@sura.ru
http://www.atlas.sura.ru

Наши реквизиты:
ИНН 771 5027275
КПП 583602001
Юр. адрес: 440601, г. Пенза, ул. Советская, 9
Р/с 40 502 810 648 000 110 518 в Пензенском ОСБ №8624 г. Пенза
К/с 30 101 810 000 000 000 635
БИК 045 655 635
ОКВЭД 73.10
ОКПО 08 859 495

ФГУП «НИИ “Квант”»

ФГУП «НИИ “Квант”» Российского агентства по системам управления является ведущим предприятием по созданию высокопроизводительных вычислительных систем, средств и систем защиты информации, комплексов автоматизации обработки информации. Направления работ НИИ «Квант» представлены в ряде государственных целевых программ и соответствуют документам «Приоритетные направления развития науки, технологий и техники Российской Федерации» и «Перечень критических технологий Российской Федерации», утвержденным Президентом РФ 30.3.02.

Основные тематические направления работ ФГУП «НИИ “Квант”»

1. Создание высокопроизводительных вычислительных систем как проблемно-ориентированных, так и широкого применения для решения научно-технических задач, требующих больших объемов вычислений и обработки данных.
2. Создание комплексов подготовки и предварительной обработки данных для последующей централизованной обработки.
3. Создание комплексов связи и документооборота на основе средств вычислительной техники с обеспечением защиты информации.
4. Объединение средств, систем и комплексов в виде территориально-распределенных информационно-вычислительных сетей.
5. Комплексная защита информации при ее обработке средствами вычислительной техники; обеспечение защиты информации в информационно-вычислительных системах и сетях.

По тематическим направлениям своей деятельности предприятие имеет уникальное опытно-экспериментальное и стендовое оборудование и высококвалифицированные кадры с большим опытом работы. Результаты научно-исследовательских и опытно-конструкторских работ ФГУП «НИИ “Квант”» и, созданные при этом, аппаратн-программные средства, системы и комплексы по технико-экономическим показателям являются передовыми в стране. Они соответствуют уровню мировых стандартов, освоены в производстве, внедрены в оперативную эксплуатацию в научных и производственных организациях страны, обеспечили решение сложных задач информатизации качественно нового уровня в интересах повышения научно-технического потенциала страны.

Работы ФГУП «НИИ “Квант”» получили признание в Росссии и среди зарубежных организаций, что подтверждено публикациями, премиями и наградами государственного уровня по основным тематическим направлениям на всех этапах работ.

ФГУП «НИИ “Квант”», как ведущее предприятие, стабильно работает и развивается на протяжении более 40 лет, обеспечивая развитие результирующих показателей в темпе мирового научно-технического прогресса по своим тематическим направлениям, соответствующим основным направлениям радиоэлектроники и средств связи.

ФГУП «НИИ “КВАНТ”»
Адрес: 125438, Москва, 4-й Лихачевский пер., 15
Тел.: 154-80-21, Тел./факс: 154-14-18
E-mail: korv@a5.kiam.ru

КРИПТО ПРО

ООО «Крипто-Про»

Краткая информация о компании Крипто-Про

Компания Крипто-Про создана в 2000 году. Основное направление деятельности компании — разработка средств криптографической защиты информации и развитие Инфраструктуры Открытых Ключей (Public Key Infrastructure) на основе использования международных рекомендаций и российских криптографических алгоритмов.

Крипто-Про имеет все необходимые лицензии ФСБ, ФАПСИ и Гостехкомисии на право осуществления деятельности по разработке, производству, распространению и техническому сопровождению шифровальных (криптографических) средств, предоставлению услуг в области шифрования информации и является аккредитованным ФАПСИ аттестационным центром в части криптографической защиты информации.

Продукты компании

Средство криптографической защиты «КриптоПро»

Средство криптографической защиты КриптоПро CSP разработано по согласованному с ФАПСИ техническому заданию в соответствии с криптографическим интерфейсом фирмы Microsoft — Cryptographic Service Provider (CSP). КриптоПро CSP имеет сертификаты соответствия ФАПСИ и может использоваться для формирования ключей шифрования и ключей электронной цифровой подписи, шифрования и имитозащиты данных, обеспечения целостности и подлинности информации, не содержащей сведений, составляющих государственную тайну.

В настоящее время КриптоПро CSP функционирует на операционных системах Windows, Solaris (Intel, Sparc) и переносится на операционные системы Linux, Free BSD, AIX.

Модуль сетевой аутентификации «КриптоПро TLS»

Модуль сетевой аутентификации КриптоПро TLS предназначен для обеспечения аутентификации клиентских и серверных компонент распределенных приложений и обеспечение конфиденциальности передаваемых данных при взаимодействии по протоколу HTTP(S).

Данный программный продукт представляет собой реализацию транспортного протокола TLS 1.0 (RFC 2246), с использованием криптографических функций СКЗИ КриптоПро CSP и сертифицирован в составе СКЗИ.

К августу 2003 года компания Крипто-Про выдала более 40 000 лицензий на использование средства криптографической защиты информации КриптоПро CSP и модуля сетевой аутентификации КриптоПро TLS, которые используется различными государственными и коммерческими организациями.

Программный комплекс «удостоверяющий центр КриптоПро УЦ»

Программный комплекс удостоверяющий центр КриптоПро УЦ разработан по согласованному с ФАПСИ техническому заданию и предназначен для обеспечения организационно-технических мероприятий при реализации функций удостоверяющего центра в соответствии с Федеральных законом «Об электронной цифровой подписи».

В состав комплекса входят компоненты, позволяющие реализовать масштабируемую, территориально распределенную инфраструктуру открытых ключей.

Программный комплекс КриптоПро УЦ успешно прошел сертификационные испытания и имеет заключение войсковой части 43753, которое подтверждает программного комплекса КриптоПро УЦ «Временным требованиям к информационной безопасности удостоверяющих центров» по классу защиты КС2.

Крипто-Про самостоятельно и совместно с системными интеграторами участвует в проектах по внедрению удостоверяющего центра КриптоПро УЦ и других продуктов компании в различных организациях, таких как:

- Комитет Российской Федерации по финансовому мониторингу;
- Аппарат Правительства Российской Федерации;
- Администрация Президента Российской Федерации;
- АС «Перепись» (Госкомстат РФ);
- Система проведения «Единого Государственного Экзамена» (Минобразования РФ);
- ОАО «Альфа Банк» в составе системы «Электронный банк “Альфа Банк Экспресс”»;
- ОАО «Альфа Банк» в составе системы «Клиент-Банк»;

- ИБГ «Никойл» в составе системы «Электронная Система Обеспечения Продаж Паев Паевых Инвестиционных Фондов»;
- ИБГ «Никойл» в составе системы «Электронный аккредитив»;

Кроме этого удостоверяющий центр КриптоПро УЦ был поставлен двадцати различным тестовой эксплуатации более чем в 30 различных организаций. Инициатива по совместимости криптографических средствВ 2001 году компания Крипто-Про разработала документ «Рекомендации к средствам криптографической защиты информации на взаимодействие удостоверяющих центров, реестров сертификатов, сертификаты ключей формата X.509 и электронные документы формата CMS», который описывает порядок применения российских криптографических алгоритмов в сертификатах открытых ключей, криптографических сообщениях, реализованных с применением продуктов компании.

В ноябре 2002 года пять ведущих российских компаний в области криптографической защиты информации подписали соглашение о сотрудничестве в области создания совместимых продуктов. В настоящее время это число увеличилось до девяти.

В рамках проходившего 57 заседания IETF (Internet Engineering Task Force) в Вене компания Крипто-Пропроекта информационных документов (Internet-Drafts), которые приняты к дальнейшему рассмотрению. Проекты документов разработаны в рамках соглашения о совместимости криптографических продуктов между перечисленными компаниями. Ознакомиться с проектами можно на сайте IETF:

`http://www.ietf.org/internet-drafts/`
`draft-leontiev-cryptopro-cpcms-00.txt`

`http://www.ietf.org/internet-drafts/`
`draft-chudov-cryptopro-cptls-00.txt`

`http://www.ietf.org/internet-drafts/`
`draft-leontiev-cryptopro-cppk-00.txt`

Кроме интеграции КриптоПро CSP в средства российских разработчиков, на сегодня две ведущих мировых производителя встроили КриптоПро CSP в свои продукты, реализующие функциональность центров сертификаций в инфраструктуре открытых ключей. К этим продуктам относятся:

- RSA Keon 6.5, производства компании RSA Security (США);
- Unicert 5.1, производства Baltimore Technologies (Ирландия);
- Работы, проводимые по заказу государственных организаций.

Крипто-Про участвует в разработках различных аппаратно-программных средств, предназначенных для обеспечения высокого уровня защиты информации при использовании технологии ЭЦП и сертификатов открытых ключей. К числу таких средств относятся:

- программно-аппаратный модуль, обеспечивающий реализацию криптографических функций и безопасное хранение ключей, предназначенный для использования в составе удостоверяющих центров;
- удостоверяющий центр, реализованный на доверенной операционной системе;
- сервисные службы удостоверяющего центра, реализованные на доверенной операционной системе.

В качестве соисполнителя в части решения вопросов информационной безопасности Крипто-Про принимает участие в работах по ФЦП «Электронная Россия». В 2002 году совместно с компанией Техносерв А/С были проведены работы по мероприятиям 15, 16, 17. В 2003 году совместно с компанией Техносерв А/С проводятся работы по лотам 5, 7.

Эксплуатация удостоверяющего центра

С июля 2001 года компания Крипто-Про предоставляет услуги юридическим лицам в качестве удостоверяющего центра (`http://ca.cryptopro.ru`), на основании сертификационного центра ФАПСИ.

На данный момент услугами центра пользуется 10 юридических лиц, в числе которых, платежная система Рапида, Вымпелком, государственная корпорация по реструктуризации кредитных организаций и другие.

127018, Москва, Улица Образцова, 38
Телефон: (095) 933 1168
Факс: (095) 933 1168
`http://www.CryptoPro.ru`
E-mail: `info@CryptoPro.ru`

Компания «РНТ»

Компания «РНТ» — один из ведущих системных интеграторов в области современных технологий безопасности.

За десять лет работы на рынке систем безопасности «РНТ» обеспечила себе прочную репутацию надежного и высокопрофессионального партнера. Компания пользуется заслуженным авторитетом среди коллег и специалистов благодаря высокому уровню проектного менеджмента, безупречному качеству и всесторонней проработке предлагаемых решений по обеспечению комплексной информационной безопасности.

Наличие полного набора лицензий на деятельность как в области безопасности информации, так и в смежных областях позволяет коллективу «РНТ» выполнять широкий комплекс работ от предпроектного обследования объекта, разработки технико-экономического обоснования и проектирования до производства строительно-монтажных работ, технического оснащения и аттестации объектов, а также их последующего сопровождения.

Компания является соучредителем ряда структур, работающих в области производства средств защиты информации и инженерно-технической защиты объектов, технической безопасности физических и юридических лиц, имеет дочерние предприятия и представительства в различных регионах России. «РНТ» является членом Российского союза промышленников и предпринимателей, Ассоциации Защиты Информации, Ассоциации документальной электросвязи, Российской инженерной академии, участником Евро-Азиатской ассоциации производителей товаров и услуг в области безопасности.

К основным направлениям деятельности компании относятся:

- Комплексная защита информации, включая применение криптографических средств защиты;
- Создание объектов информатизации в защищенном исполнении;
- Разработка средств и технологий защиты информации;
- Сертификационные испытания средств защиты информации;
- Аттестация объектов информатизации;
- Информационная безопасность систем и сетей связи;
- Защита информационно-телекоммуникационных систем;
- Аудит информационной безопасности объектов информатизации;
- Инженерная и инженерно-техническая защита объектов;
- Стрелковые тиры.

Среди заказчиков «РНТ» — Министерство обороны РФ, Министерство внутренних дел РФ, Министерство иностранных дел РФ, Гостехкомиссия России, ФСБ (ФАПСИ), Государственный таможенный комитет РФ, Министерство финансов РФ, Федеральное казначейство РФ, Комитет РФ по финансовому мониторингу, Министерство РФ по налогам и сборам, Министерство путей сообщения РФ, Министерство экономического развития и торговли РФ, Пенсионный фонд РФ, ФГУП «РОСОБОРОНЭКСПОРТ», Госатомнадзор России, Центральный банк РФ, АО «МГТС», «Росэкспертиза», ОАО «Транснефть», ОАО «НК Роснефть» и многие другие.

Коллектив компании — это ведущие специалисты в области безопасности информации, многие из которых имеют научные степени и звания, являются членами Академии криптографии РФ, Российской инженерной Академии, авторами передовых научных идей и практических разработок, многие из которых активно применяются сегодня в области создания информационно-телекоммуникационных систем, систем связи и информационной безопасности, оптических технологий.

Россия, Москва, Дмитровское шоссе, д. 2
Тел.: (095) 777 7577 (многоканальный)
Факс: (095) 777 7576
E-mail: rnt@rnt.ru
http://www.rnt.ru

СИСТЕМАТИКА
со знанием дела

О компании

Компания «Систематика» работает на рынке консалтинга и системной интеграции, предоставляя крупнейшим корпоративным Заказчикам комплекс инфраструктурных и информационных решений, позволяющих реализовывать стоящие перед ними задачи.

«Систематика» образована в результате объединения интеграторских бизнесов ИТ-холдинга «Национальная компьютерная корпорация» (НКК), а именно компании «Аквариус Консалтинг» и компании «Аквариус Дата». Расширение сферы деятельности и спектра услуг, объединение материальных, финансовых и кадровых ресурсов позволяет объединенному системному интегратору реализовать инфраструктурные и информационные решения любой степени сложности.

Компетенция компании «Систематика» подтверждена позитивными отзывами Заказчиков — крупнейших российских государственных структур и промышленных предприятий.

Основные направления деятельности

На сегодняшний день «Систематика» предлагает широкий спектр услуг, в том числе:

Консалтинг

- ИТ-консалтинг;
- разработка ИТ-стратегии.

Комплексные системы управления

- Корпоративные системы (ERP), решения по управлению проектами, основными фондами предприятия (EAM) и капитальным строительством;
- Системы управления информационно-технологическими ресурсами предприятия (ITSM);
- Интегрированные системы управления зданиями и инженерными системами;
- Системы управления жизненным циклом документов.

Инфраструктурные решения

- Инженерные системы;
- Комплексные системы обработки и хранения данных, катастрофоустойчивые ЦОД;
- Решения в области создания распределенных систем хранения данных;
- Системы мониторинга и управления сетевой инфраструктурой;
- Системы информационной и технической безопасности.

Отраслевые решения

- Автоматизированные системы управления технологическими процессами приема, хранения и отгрузки нефтепродуктов;
- Управление процессами производственного планирования в металлургии, машиностроении, оборонной и других отраслях промышленности;
- Создание порталов органов власти, регистров недвижимости, социальных регистров населения;
- Отраслевые решения на базе ERP-систем.

Аутсорсинг и сервисная поддержка

Комплексные поставки вычислительной техники

Мы постоянно наращиваем компетенции, развиваем новые направления, разрабатываем новые информационные решения, востребованные рынком.

Россия, 121609, Москва, Осенний бульвар, 23.
Тел.: (095) 729-51-99, факс: (095) 729-59-80, e-mail: info@sys.ncc.ru,
http:\\www.systematic.ru

ЗАО «С-Петербургский РЦЗИ»

ЗАО «Санкт-Петербургский Региональный центр защиты информации» (СПб РЦЗИ) образован в 1997 году.

Санкт-Петербургский РЦЗИ предоставляет следующие виды услуг:

- Подготовка и построение сетей защищенной передачи информации, соответствующих требованиям Гостехкомиссии РФ, ФАПСИ, предъявляемых к конфиденциальной информации и информации, составляющей Государственную тайну (гриф «совершенно секретно» и «секретно»).
- Поставка комплекса оборудования криптографической защиты информации, имеющего необходимые сертификаты Российской Федерации, включающем аппаратно-программный комплекс организации цифровой подписи (ЦП) в сетях электронной торговли, автоматизированной бухгалтерии и др.

В настоящее время Санкт-Петербургским Региональным Центром Защиты информации разработаны, сертифицированы ФАПСИ и широко внедряются комплексы средств криптографической защиты на базе малогабаритных приборов и Российских интеллектуальных карт, обеспечивающие защиту в каналах связи информации с грифом секретности «конфиденциально» (М 484К), «секретно» (М 484КС) и «сов. секретно» (М 484КМ, М 448). Комплексы имеют высокие эксплуатационные характеристики и получили положительную оценку эксплуатирующих организаций ФСО, ФСБ, МО, ГУИН.

Основными особенностями комплексов являются:

- Не секретность аппаратуры в не активизированном состоянии.
- Независимость от программно-аппаратных средств информационно-вычислительных систем, возможность использования с широким спектром программных и аппаратных средств развернутых систем.
- Широкий спектр отработанных вариантов создания защищенного тракта на базе стандартных коммуникационных средств.

Для использования в системах VISA и MasterCard разработан и сертифицирован прибор РМ (Российский модуль безопасности), который реализует функции прибора типа HSM серии 7000, а так же дополнительный контур защиты на основе российских криптографических стандартов.

Телефон (812) 590 7381, факс (812) 590 7381,
E-mail: `erkin@nevsky.net`, `http://www.rczi.spb.ru`