Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Оренбургский государственный университет»

Н.А. Тишина

ЗАЩИТА ИНФОРМАЦИОННЫХ ПРОЦЕССОВ В КОМПЬЮТЕРНЫХ СИСТЕМАХ

Учебное пособие

Рекомендовано ученым советом федерального государственного бюджетного образовательного учреждения высшего образования «Оренбургский государственный университет» для обучающихся по образовательным программам высшего образования по направлениям подготовки 09.03.01 Информатика и вычислительная техника и 09.03.04 Программная инженерия

Оренбург
2019

УДК 004.4 : 378.245 (076.5)
ББК 32.973-018.2я7+74.58я7
Т 60

Рецензент – кандидат технических наук, доцент В.Н. Костин

Тишина, Н.А.
Т 60 Защита информационных процессов в компьютерных системах: учебное пособие / Н.А.Тишина; Оренбургский гос. ун-т. – Оренбург: ОГУ, 2019. – 179 с.

ISBN

В учебном пособии содержатся материалы, необходимые студентам для изучения дисциплины «Защита информационных процессов в компьютерных системах». В разделах учебного пособия рассматриваются: концепции и термины в области компьютерной безопасности; некоторые фундаментальные концепции и технические детали криптографии; основные понятия аутентификации, детальное изложение концепции строгой аутентификации и некоторых протоколов её реализации; обзор, классификация и технические сведения программно-аппаратных средств защиты информации; вопросы безопасности распределенных приложений.

Учебное пособие предназначено для обучающихся по направлениям подготовки 09.03.01 Информатика и вычислительная техника, 09.03.04 Программная инженерия для изучения дисциплины «Защита информационных процессов в компьютерных системах».

УДК 004.4 : 378.245 (076.5)
ББК 32.973-018.2Я7+74.5Я7

ISBN

Содержание

Введение

Защита (обеспечение) безопасности информационных процессов компьютерных систем (КС) является одной из главных функций КС при их создании и эксплуатации. Расширяются области применения информационных технологий, усложняется инфраструктура, совершенствуются технологии. Эти тенденции являясь фактором развития, одновременно порождают новые информационные угрозы, следовательно, ведут к необходимости развития теории и практики защиты компьютерной информации.

Необходимость совершенствования защиты КС, представляя одну из основных проблем современного мира из-за ее сложности, и с точки зрения политики и с точки зрения технологии вызывает рост потребности в более широких знаниях в области компьютерной безопасности в обществе, а также в квалифицированных специалистах в области информационных технологий.

Следовательно, к основным задачам становления специалиста в области информационных технологий добавляются:

- формирования навыков определения основных угроз информации в КС;
- освоение принципов безопасности информационных процессов КС;
- освоение методов, моделей и средств защиты информационных процессов КС;
- формирования умений и навыков разработки и применения программных средств (ПС) защиты информации (ЗИ) в процессе ее сбора, хранения, обработки, передачи и распространения в КС.

В данном учебном пособии содержатся материалы, необходимые студентам, обучающимся по направлениям подготовки 09.03.01 Информатика и вычислительная техника, 09.03.04 Программная инженерия для изучения дисциплины «Защита информационных процессов в компьютерных системах».

В первом разделе учебного пособия приводятся основные концепции и термины в области компьютерной безопасности. Формулирование терминов

опирается на законы РФ, стандарты и нормативные документы в области информационной безопасности.

Во втором разделе рассматриваются некоторые фундаментальные концепции и технические детали криптографии. Например, фактический процесс некоторых криптографических алгоритмов. Но основное внимание уделяется широкому пониманию соответствующих концепций. Раздел предназначен для того, чтобы дать общее представление о том, что такое шифрование, как оно работает, и достаточно информации для принятия разумного решения о том, какие типы шифрования должны быть использованы в организации.

В третьем разделе рассматриваются основные понятия аутентификации. Приведено детальное изложение концепции строгой аутентификации и некоторых протоколов её реализации.

В четвертом разделе приведены обзор и классификация программно-аппаратных средств ЗИ в КС, таких как антивирусы, межсетевые экраны, VPN, системы обнаружения вторжений, приведены протоколы безопасности Wi-Fi. Излагаются технические сведения о средствах достаточно подробно, чтобы принимать разумные решения о том, какие типы средств следует применять.

В пятом разделе рассматриваются вопросы безопасности распределенных приложений, некоторые текущие тенденции в области информационных технологий и проблемы безопасности, которые могут возникнуть, а также предоставлены некоторые окончательные соображения по обеспечению безопасности.

1 Основные понятия информационной безопасности

1.1 Информационная безопасность и кибербезопасность

Современные информационные технологии (ИТ) охватывают все сферы жизнедеятельности человека, формируя единое информационное пространство.

Все более растет потребность в КС, Интернет и беспроводных сетях, таких как Bluetooth и Wi-Fi, и рост "умных" устройств, таких как смартфоны, телевизоры и различные портативные устройства, составляющих Интернет Вещей. С помощью глобальной вычислительной сети Интернет объединяются и перемещаются на любые расстояния гигантские объемы информации, обеспечивается доступ многочисленных пользователей, расположенных на практически неограниченной территории, к информационным ресурсам всего мирового сообщества.

Вместе с тем, компьютерами и сетями злоупотребляют с растущей скоростью. Спам, фишинг и компьютерные вирусы становятся многомиллиардными проблемами, как и кража личных данных, что создает серьезную угрозу для личных финансов и кредитных рейтингов пользователей и создает обязательства для корпораций.

Расширяются области применения информационных технологий, усложняется инфраструктура, совершенствуются технологии. Эти тенденции являясь фактором развития, одновременно порождают новые информационные угрозы, следовательно, ведут к необходимости совершенствования защиты КС, представляя одну из основных проблем современного мира из-за ее сложности, и с точки зрения политики и с точки зрения технологии.

Таким образом, растет потребность в более широких знаниях в области компьютерной безопасности в обществе, а также в повышении квалификации специалистов в области ИТ.

Одной из первых вещей, которые необходимо сделать в учебном пособии по компьютерной безопасности, является определение концепций и терминов.

Необходимо *различать следующие понятия*: информационная безопасность, безопасность информации (Information security), компьютерная безопасность,

кибербезопасность или безопасность информационной технологий (безопасность IT-систем). Отличие понятий заключается в *объекте защиты*. В основе этих понятий лежит *информационная безопасность* (ИБ). Методологической базой ИБ является понятие безопасности [25].

ИБ гарантирует, что и физические, и цифровые данные будут защищены от несанкционированного доступа, использования, раскрытия, разрушения, модификации, контроля, записи или разрушения. Информационная безопасность в отличие от кибербезопасности стремится сохранять безопасность данных в любой форме, тогда как кибербезопасность защищает только цифровые данные.

Компьютерная безопасность (КБ), *кибербезопасность* или *безопасность информационных технологий* (безопасность IT-систем) (information technology security) представляет подмножество ИБ, обеспечивает защиту компьютерных систем, сетей и данных от несанкционированного цифрового доступа, кибератак или повреждения, реализуя различные процессы, технологии и средства. С бесчисленными сложными агентами угроз, предназначающимися для всех типов организаций, критически важно, чтобы инфраструктура ИТ была защищена в любой ситуации, чтобы была обеспечена возможность предотвращения полномасштабных атак на сеть и рисков нарушения данных и репутации.

Эффективная кибербезопасность снижает риск кибератак и защищает организации и людей от несанкционированной эксплуатации систем, сетей и технологий.

Кибератаки могут разрушить и нанести значительный финансовый и репутационный ущерб даже самой высокотехничной организации. Кибератака может привести к потере активов, репутации и бизнеса, к судебным тяжбам, а также затратам на восстановление.

ИБ является компонентой национальной безопасности наряду с общественной, личной, экологической и др. видами [33] (рисунок 1.1).

Под информационной безопасностью Российской Федерации согласно документу «ДОКТРИНА информационной безопасности Российской Федерации», утвержд. Указом Президента Российской Федерации от 5 декабря 2016 г. №646:

«*информационная безопасность Российской Федерации* (далее – информационная безопасность) – состояние защищенности личности, общества и государства от внутренних и внешних информационных угроз, при котором обеспечиваются реализация конституционных прав и свобод человека и гражданина, достойные качество и уровень жизни граждан, суверенитет, территориальная целостность и устойчивое социально-экономическое развитие Российской Федерации, оборона и безопасность государства» [10].

Рисунок 1.1 – Место информационной безопасности в общей системе национальной безопасности России

К *объектам ИБ* относятся [14]:

– информационные ресурсы;

– права граждан, юридических лиц и государства на получение, распространение и использование информации, защиту конфиденциальной информации и интеллектуальной собственности;

– системы формирования, распространения и использования информационных ресурсов;

– информационная инфраструктура;

– системы формирования общественного сознания.

Компьютерная безопасность (КБ) – состояние защищенности (безопасность) информации в компьютерных системах и безотказность (надежность) функционирования компьютерных систем [4].

К *объектам КБ* относятся [36]: информация и программно-технический комплекс.

Безопасность информации (БИ) является частью компьютерной безопасности и трактуется согласно документу «Руководящий документ. Защита от несанкционированного доступа к информации. Термины и определения», утвержденному решением председателя Гостехкомиссии России от 30 марта 1992 г как «*безопасность информации* (Information security) это состояние защищенности информации, обрабатываемой средствами вычислительной техники или автоматизированной системы, от внутренних или внешних угроз» [37].

1.2 Компьютерная система как объект защиты информации

ЗИ предполагает знание объекта защиты, основ построения компьютерных систем и перечень основных компонент.

Современная КС по своему назначению и содержательной сущности равно, как и информационная система (ИС), представляет собой взаимосвязанную совокупность содержащейся в базах данных информации и обеспечивающих ее обработку информационных технологий и технических средств в интересах достижения поставленной цели [26].

Функциональной основой КС являются информационные процессы, протекающие в ней. Характер этих процессов определяется соответствующей ИТ. ИТ (Статья 2 ФЗ №149 от 27.07.2006) – это «процессы, методы поиска, сбора, хранения, обработки, предоставления, распространения информации и способы

осуществления таких процессов и методов» [26]. Иначе говоря, информационные процессы КС, являясь частью ИТ, представляют собой процессы сбора, обработки, накопления, хранения, поиска и распространения информации.

КС *как объект защиты* представляет собой совокупность следующих взаимосвязанных компонентов:

– данных – хранимых временно и постоянно, на магнитных носителях, печатные, архивы, системные журналы и т.д.;

– технических средств обработки и передачи данных (компьютерных и телекоммуникационных средств);

– программных средств, реализующих соответствующую технологию обработки информации;

– обслуживающего персонала и пользователей системы, объединенных по организационному, предметно-тематическому, технологическому и другим принципам для выполнения автоматизированной обработки информации (данных).

– в более широком смысле включает также помещения, здания, а также прилегающую к зданиям территорию.

Классификация компьютерных систем

Классификация компьютерных систем может проводиться по различным признакам, таким как [22, 28]:

- способ построения;
- функциональное назначение;
- размещение информации в сети;
- число главных вычислительных машин (ГВМ);
- тип используемых ЭВМ;
- метод передачи данных;
- реализация топологии соединения компьютерных систем в сети.

По способу построения, подразделяют на сосредоточенные и распределенные [28].

По функциональному назначению различают КС: автоматизированной обработки данных (АСОД) и автоматизированные системы контроля управления производством, технологическими процессами и объектами (АСК). АСОД делят на:

– информационные, предоставляющие пользователю в основном информационное обслуживание;

– вычислительные, выполняющие главным образом решение задач с обменом данными и программами между ЭВМ сети,

– смешанные информационно-вычислительные.

Подробнее описание по другим признакам можно посмотреть в [22, 28]

1.3 Виды защищаемой информации

Информация согласно Федеральному закону от 27.07.2006 № 149-ФЗ (ред. от 23.04.2018) "Об информации, информационных технологиях и о защите информации" представляет собой «сведения (сообщения, данные) независимо от формы их представления» [26].

Возникает вопрос: «Всю ли информацию нужно защищать?». Ответ на этот вопрос определяется ценностью (определяется степенью её полезности [21, стр. 22-24]), значимостью информации, а также законами и нормативными документами.

К защищаемой информации относится «информация, являющаяся предметом собственности и подлежащая защите в соответствии с требованиями правовых документов или требованиями, устанавливаемыми собственником информации» [33].

«Уровень значимости информации определяется степенью возможного ущерба для обладателя информации (заказчика) и (или) оператора от нарушения конфиденциальности, целостности или доступности информации» [34]:

«УЗ = [(конфиденциальность, степень ущерба) (целостность, степень ущерба) (доступность, степень ущерба)],

где степень возможного ущерба определяется обладателем информации (заказчиком) и (или) оператором самостоятельно экспертным или иными методами».

Защите подлежит информация ограниченного доступа, это определено ФЗ №149 от 27.07.2006, который подразделяет информацию в зависимости от категории доступа к ней на *общедоступную* и *информацию ограниченного доступа* [26].

К информации ограниченного доступа относится (ст. 9 ФЗ №149) [26]:

- государственная тайна;
- коммерческая тайна;
- служебная тайна;
- профессиональная тайна;
- личная или семейная тайна, гражданина (физического лица), информация о его частной жизни;
- персональные данные граждан (физических лиц).

Уточненный список конфиденциальной информации можно посмотреть в документе «Перечень сведений конфиденциального характера», утвержденном Указом Президента Российской Федерации от 6 марта 1997 г. № 188 [27], свод перечня с комментариями в [1].

Также выделяют информацию, к которой *не может быть ограничен доступ* (см. ст. 8 ФЗ №149).

1.4 Конфиденциальность, целостность и доступность

Классически информационная безопасность была определена в терминах акронима C.I.A., означающего *конфиденциальность, целостность и доступность*. Также и "безопасность информации" включает три составляющих:

- обеспечение конфиденциальности;
- обеспечение целостности;
- обеспечение доступности.

Под *конфиденциальностью* информации понимается «специфическое свойство отдельных категорий (видов) информации, которое субъективно устанавливается ее обладателем, когда ему может быть причинен ущерб от

ознакомления с информацией неуполномоченных на то лиц, при условии того, что обладатель принимает меры по организации доступа к информации только уполномоченных лиц» [4].

В контексте компьютерной безопасности конфиденциальность заключается в предотвращении несанкционированного раскрытия информации. То есть конфиденциальность включает в себя защиту данных, обеспечивая доступ тем, кому разрешено видеть данные, не позволяя другим узнать что-либо об их содержании.

Сохранение секретности информации часто лежит в основе информационной безопасности, и эта концепция, по сути, предшествует компьютерам. Например, шифр Юлия Цезаря. В настоящее время достижение конфиденциальности является более сложной задачей. Компьютеры повсюду, и каждый из них способен выполнять операции, которые могут поставить под угрозу конфиденциальность. Поэтому исследователи компьютерной безопасности и разработчики КС разработали ряд *инструментов для защиты конфиденциальной информации*. Эти инструменты включают в себя следующие концепции:

Шифрование: преобразование информации с помощью секретного ключа, называемого ключом шифрования, так что преобразованная информация может быть прочитана только с использованием другого секретного ключа, называемого ключом дешифрования (который в некоторых случаях может быть таким же, как ключ шифрования). Для безопасности, схема шифрования должна затруднить кому-либо определение исходной информации без использования ключа дешифрования.

Контроль доступа: правила и политики, которые ограничивают доступ к конфиденциальной информации людям и / или системам с «необходимостью узнать эту информацию». Доступ реализуется с помощью идентификации, например, имени человека или серийного номера компьютера или роли, которую человек имеет, например, быть менеджером или специалистом по компьютерной безопасности.

Аутентификация: определение личности или роли, которой обладает кто-то. Определение может быть сделано несколькими способами, но обычно оно основано на комбинации нескольких факторов: того, что у человека имеется (например,

смарт-карты или фан-памяти радио, хранящей секретные ключи), что-то, что человек знает (например, пароль) и что-то такое, что человек (как человек с отпечатком пальца).

Авторизация: определение того, что разрешен ли человеку или системе доступ к ресурсам на основе политики контроля доступа. Такие разрешения должны препятствовать злоумышленнику обманывать систему, позволяя ему иметь доступ к защищенным ресурсам.

Под *целостностью* информации (данных) понимается «неискаженность, достоверность, полнота, адекватность и т.д. информации, т.е. такое ее свойство, при котором содержание и структура данных определены и изменяются только уполномоченными лицами и процессами» [4].

Ранее упомянутые инструменты защиты конфиденциальности информации, запрещающие доступ к данным пользователям без соответствующих прав доступа, также помогают предотвратить изменение данных в первую очередь. Кроме того, существует несколько *инструментов*, специально предназначенных *для обеспечения целостности*:

Резервные копии: периодическое архивирование данных. Это архивирование выполняется таким образом, что файлы данных могут быть восстановлены, если они когда-либо будут изменены неавторизованным или непреднамеренным способом.

Контрольные суммы: вычисление значений функции, которая отображает содержимое файла в числовое значение. Функция контрольной суммы зависит от всего содержимого файла и разработана таким образом, что даже небольшое изменение во входном файле (например, переворачивание одного бита), скорее всего, приведет к другому выходному значению. Контрольные суммы используются для выявления случаев нарушения целостности данных.

Коды коррекции данных: способы хранения данных таким образом, что небольшие изменения могут быть легко обнаружены и автоматически исправлены. Эти коды обычно применяются к небольшим единицам хранения (например, на уровне байта или слота памяти), но есть также коды коррекции данных, которые также могут применяться ко всем файлам.

Эти инструменты для обеспечения целостности данных имеют общую черту – они используют избыточность. То есть они включают в себя репликацию некоторого информационного контента или характеристик данных, чтобы можно было обнаружить и иногда даже исправить нарушения целостности данных. Кроме содержания файла данных необходимо поддерживать в отношении целостности *метаданные* для каждого файла данных, которые являются атрибутами файла или информацией о доступе к файлу, которые не являются частью его содержимого. Примеры метаданных включают пользователя, который является владельцем файла, последнего пользователя, который изменил файл, последнего пользователя, который прочитал файл, даты и время создания и последнего изменения файла, имя и расположение файла в файловой системе и список пользователей или групп, которые могут читать или записывать файл. Таким образом, изменение любых метаданных файла должно считаться нарушением его целостности. Например, злоумышленник компьютера не может фактически изменять содержимое любых пользовательских файлов в системе, в которую он проник, но он тем не менее может изменять метаданные, такие как метки времени доступа, просматривая файлы (и тем самым нарушая их конфиденциальность, если они не зашифрованы). Действительно, если система выполнит проверку целостности для этого типа метаданных, она может обнаружить вторжение, которое иначе осталось бы незамеченным.

«Под [правомерной] *доступностью* информации (данных) понимается такое свойство информации, при котором отсутствуют препятствия доступа к информации и закономерному ее использованию обладателем или уполномоченными лицами» [4].

Действительно, качество некоторой информации напрямую связано с тем, насколько она доступна. Например, котировки акций наиболее полезны, когда они свежи. Кроме того, представьте себе ущерб, который может быть нанесен, при краже кредитной карты, если потребуется несколько недель, прежде чем компания по кредитным картам сможет уведомить кого-либо, потому что ее список украденных номеров был недоступен.

Существует ряд *инструментов для обеспечения доступности*, в том числе следующие:

Физическая защита: инфраструктура, предназначенная для обеспечения доступности информации даже в случае физических проблем. Такая защита может включать в себя здания, в которых могут быть построены критически важные КС, такие чтобы выдерживать штормы, землетрясения и взрывы бомб, а также оснащаться генераторами и другим электронным оборудованием, чтобы справляться с перебоями в подаче электроэнергии и перенапряжениями.

Вычислительные избыточности: компьютеры и устройства хранения данных, которые служат резервами в случае сбоев. Например, избыточные массивы дисков (RAID) используют избыточность хранилищ для хранения данных для своих клиентов. Кроме того, веб-серверы часто организованы в виде мультипликаторов, называемых «фермами», так что сбой любого компьютера может быть устранен без ухудшения доступности веб-сайта. Поскольку доступность настолько важна, злоумышленник, который не заботится о конфиденциальности или целостности данных, может решить атаковать его доступность. Например, вор, который крадет сразу много кредитных карт, может атаковать доступность списка украденных кредитных карт, которые поддерживаются и транслируются крупной компанией с кредитными картами.

Таким образом, обеспечение доступности является третьим этапом поддержки жизненно важной C.I.A., триады информационной безопасности.

1.5 Уверенность, достоверность и анонимность

На ряду с классической концепцией C.I.A. конфиденциальности, целостности и доступности, существует ряд *дополнительных концепций*, которые также важны в современных приложениях компьютерной безопасности. Эти понятия могут быть также охарактеризованы *трехбуквенным акронимом A.A.A.* (Assurance, Authenticity, and Anonymity) означающим уверенность, достоверность и анонимность [44, 48].

Уверенность.

В отличие от C.I.A. концепций, понятия A.A.A. не зависят друг от друга. Уверенность, в контексте компьютерной безопасности, связано с обеспечением доверия и его управлением в КС. Доверие сложно оценить количественно, но оно связано с уверенностью в том, что люди или системы ведут себя так, как от них ожидают.

Кроме того, *доверие включает* в себя следующее:

– *Политики* определяют правила поведения людей или систем. Например, разработчики онлайновой музыкальной системы могут применять политики, описывающие, как пользователи могут получать и копировать песни.

– *Разрешения* описывают поведение, разрешенное программными агентами, взаимодействующими с человеком или системой. Например, интернет-магазин музыки может предоставлять разрешения для ограниченного доступа и копирования для людей, которые приобрели определенные песни.

– *Защиты* описывают механизмы обеспечения прав и полномочий. Например, для онлайн-магазина музыки, можно создать защиту от несанкционированного доступа и копирования своих песен.

Однако уверенность не просто направлена от систем к пользователям.

Пользователь, предоставляющий номер своей кредитной карты в онлайн-музыкальной системе, может ожидать, что система будет соблюдать свои опубликованные политики в отношении использования номеров кредитных карт, может предоставить разрешение системе начислять небольшие сборы за свою карточку за покупки музыки, ожидать, что система имеет защиту кредитных карт от мошеннических действий по карточке.

Таким образом, в отношении КС уверенность заключается в управлении доверием по двум направлениям: от пользователей к системам и от систем до пользователей.

Разработчики КС хотят защитить не только конфиденциальность, целостность и доступность информации. Они также хотят защитить и управлять ресурсами систем, и чтобы пользователи не злоупотребляли этими ресурсами. Например, чтобы несанкционированные пользователи не использовали их процессоры, память

и сети, даже если никакая информация не будет скомпрометирована с точки зрения C.I.A. концепции.

Таким образом, разработчики КС хотят, чтобы люди, использующие ресурсы их систем, делали это в соответствии с их политикой. Аналогичным образом, управление информацией в КС также может выходить за пределы C.I.A. в котором, возможно, нужно *управлять использованием информации*. Например, если пользователь системы онлайн-проката фильмов арендовал электронную копию фильма, мы, можем позволить этому пользователю смотреть его только фиксированное количество раз, или чтобы он смотрел его в течении только 30 дней.

Таким образом, управление доверием связано с разработкой эффективных политик, обеспечивающих предоставление разрешений доверенным пользователям и компонентам, для защиты и управления ресурсами в системе.

Лучше всего, если будут разработаны политики, которые легко соблюдать и выполнять.

Еще одна важная часть обеспечения безопасности КС – разработка программного обеспечения (ПО).

Проектировщики КС должны контролировать реализацию программного обеспечения системы в *соответствии их проекту*. На самом деле существует множество примеров систем, которые были правильно спроектированы, но которые работали некорректно, потому что эти проекты реализованы неправильно. Классическим примером такой неправильной реализации является использование генераторов псевдослучайных чисел.

Генератор псевдослучайных чисел (PRNG) представляет собой программу, которая возвращает последовательность чисел, которые являются статистически случайными, с учетом *начального числа*, называемого *зерном*, которое предполагается случайным. Проектировщик системы может указать, что PRNG будет использоваться в определенном контексте, например, для шифрования, чтобы каждый шифр был разным. Но программист совершает ошибку, всегда используя одно и то же зерно для генератора псевдослучайных чисел, тогда

последовательности псевдослучайных чисел всегда будут одинаковыми, а шифр со слабой криптостойкостью.

Таким образом, разработчики безопасных систем должны иметь не только хорошие проекты, но также должны иметь хорошие *спецификации и реализации*. Обеспечение доверия от пользователей к системе более проблематично. Обычно пользователи не имеют той же вычислительной мощности, что и серверы, используемые такими системами. Таким образом, доверие пользователей к системе должно исходить из ограниченного количества вычислений, которые они могут выполнять, а также от юридического и репутационного ущерба, который пользователь может причинить компании, владеющей системой, если она не сможет соответствовать доверию пользователя.

Достоверность

Большое количество онлайн-сервисов, предоставляющих контент, ресурсы и даже вычислительные сервисы, привело к необходимости применения этими системами своих политик. Юридически это требует, наличие электронного способа обеспечения исполнения контрактов. То есть, когда кто-то намерен купить песню в онлайн-магазине музыки, должен быть какой-то способ обеспечить это обязательство. Аналогичным образом, когда онлайн-магазин фильмов разрешает пользователю арендовать фильм и смотреть его где-то в течение следующих 30 дней, у этого пользователя должен быть какой-то принудительный способ узнать, что фильм будет доступен в течение всего этого времени.

Подлинность – это способность определить истинность утверждений, политик и разрешений, выданных людьми или системами. Если такие вещи могут быть сфальсифицированы, то нет способа обеспечить соблюдение подразумеваемых контрактов, в которых участвуют люди и системы при покупке и продаже товаров через Интернет. Кроме того, человек или система могут утверждать, что они не брали на себя такое обязательство – они могут сказать, что обязательство было принято кем-то, притворяющимся им. Формально говорят, что протокол, который обеспечивает такие типы аутентичности, демонстрирует невозможность опровержения, *неопровержимость или неотказность.*

Неопровержимость – это свойство, с которым нельзя отрицать подлинные заявления, сделанные каким-либо лицом или системой. Главный способ, которым достигается свойство неотказности – это использование *цифровых подписей*. Это криптографические вычисления, которые позволяют человеку или системе принять на себя обязательство аутентичности своих документов уникальным способом, который достигает неопровержимости. Фактически, цифровые подписи обычно имеют некоторые дополнительные преимущества по сравнению с рукописными подписями, поскольку цифровые подписи также позволяют проверять целостность подписанных документов. То есть, если документ изменен, подпись на этом документе становится недействительной.

Поэтому важным требованием к подлинности является наличие надежных способов электронной идентификации людей.

Анонимность.

Как правило, люди взаимодействуют с системами через их *реальные идентификационные данные*, что может привести к негативному побочному эффекту. В конечном итоге идентификационные данные распространяются на множество цифровых записей, что связывает эту личность с её историей болезни, историей покупки, юридическими документами, электронными сообщениями, записями занятости и т. д. Поэтому возникает потребность в *анонимности*, когда определенные записи или транзакции не будут относиться к какому-либо лицу. Организации должны *публиковать данные* о своих членах или клиентах с сохранением *конфиденциальности*, используя некоторые из следующих *инструментов*:

– *Агрегация*: объединение данных от многих людей, так что раскрытые суммы или средние значения не могут быть привязаны к какому-либо лицу. Например, перепись населения США регулярно публикует демографические данные по регионам, почтовым индексам, этническому признаку, зарплате, возрасту и т. д.

– *Смешивание*: перемешивание транзакций, информации или сообщений таким образом, чтобы никто не мог отследить. Этот метод несколько технический, но он включает в себя системы, которые могут смешивать данные вместе квази-

случайным образом, чтобы транзакции или поисковые запросы все равно могли выполняться, но без раскрытия какой-либо отдельной личности.

– *Прокси*: доверенные агенты, участвующие в действиях человека таким образом, что не может быть прослежен путь до этого человека. Например, поисковые прокси-сайты в Интернете – это веб-сайты, которые предоставляют интерфейс интернет-браузера, так что люди могут посещать веб-сайты, заблокированные, например, в стране, в которой они находятся.

– *Псевдонимы*: вымышленные идентификаторы, которые могут заполняться для реальных субъектов в сообщениях и транзакциях, но реальные идентификаторы известны только доверенным субъектам. Например, многие онлайн-сайты социальных сетей позволяют пользователям взаимодействовать друг с другом с использованием псевдонимов, чтобы они могли общаться и создавать онлайн-персону, не раскрывая их фактическую идентичность.

1.6 Угроза, уязвимость, атака

Обсудив различные цели компьютерной безопасности, следует рассмотреть понятия уязвимости, угрозы и атаки, которые могут нарушить эти цели:

«*Уязвимость компьютерной системы* – некая ее характеристика, делающая возможным возникновение угрозы» [33].

На рисунке 1.2 приведена классификация *уязвимостей программного обеспечения* согласно документу [33].

Угроза безопасности информации согласно ГОСТ Р 50922-2006 – «совокупность условий и факторов, создающих потенциальную или реально существующую опасность нарушения безопасности информации» [6].

Источниками угроз могут выступать люди – внутренние и внешние нарушители, вредоносное программное обеспечение, неполадки и катастрофы. Подробную классификацию программных вирусов и червей по различным критериям, а также классификацию угроз БИ с использованием протоколов межсетевого взаимодействия можно посмотреть в документе [33].

Рисунок 1.2 – Классификация уязвимостей ПО

Атака на компьютерную систему – действие, предпринимаемое злоумышленником, которое заключается в поиске и использовании той или иной уязвимости.

С атакой связано также понятие *окна опасности*, представляющего собой промежуток времени от момента появления возможности использования уязвимости КС, и до момента её ликвидации. Пока существует окно опасности, возможны успешные атаки на КС.

В контексте ошибок ПО окно опасности начинается с появлением средств использования ошибки и заканчивается после установки заплат, ее исправляющих. Занимает это довольно много времени (несколько дней или недель), так как:

– должно стать известно о средствах использования уязвимости в защите;

– должны быть выпущены и установлены соответствующие заплаты в защищаемой КС.

Некоторые из угроз и атак, которые могут поставить под угрозу БИ:

– *Подслушивание*: перехват информации, предназначенной для кого-то еще во время ее передачи по каналу связи. Примеры включают пакетные снифферы, которые контролируют соседний Интернет-трафик, например, в точке беспроводного доступа. Подслушивание является атакой на конфиденциальность.

– *Изменение*: несанкционированное изменение информации. Примерами атак с изменением являются атака «человек-в-середине», где сетевой поток перехватывается, модифицируется и повторно передается, а также компьютерные вирусы, которые изменяют критические системные файлы, чтобы выполнять некоторые вредоносные действия и реплицировать себя. Изменение является атакой на целостность данных.

– *Отказ в обслуживании*: прерывание или деградация службы данных или доступа к информации. Например, спам электронной почты, предназначенный для простого заполнения очереди сообщений и замедления работы почтового сервера. Отказ в обслуживании - это атака на доступность.

– *Маскарад*: изготовление информации, которая, как предполагается, принадлежит тому, кто на самом деле не является автором. Примеры маскарадных атак включают фишинг, который создает веб-сайт, который выглядит как настоящий банк или другой сайт электронной коммерции, но предназначен только для сбора паролей и подмены, что может включать отправку по сетевым пакетам данных, которые имеют ложные обратные адреса. Маскарад - это нападение на подлинность, а в случае фишинга – попытка скомпрометировать конфиденциальность и / или анонимность.

– *Отказ (ренегатство)*: отказ в обязательстве или получении данных. Это предполагает попытку отказаться от контракта или протокола, который требует от разных сторон предоставления квитанций, подтверждающих, что данные получены. Это нападение на уверенность.

– *Корреляция и отслеживание*: интеграция нескольких источников данных и информационных потоков для определения источника конкретного потока данных или части информации. Это нападение на анонимность.

Существуют и другие виды атак, такие как атаки на военном уровне, предназначенные для раскрытия криптографических секретов. Кроме того, есть составные атаки, которые объединяют несколько вышеупомянутых типов атак в один. Но перечисленные выше являются одними из наиболее распространенных типов атак.

Также важными источниками информации об угрозах, уязвимостях, атаках являются различные базы (банки) данных (знаний), такие как:

– Банк данных угроз безопасности информации ФСТЭК РОССИИ Федеральной службы технического и экспертного контроля, расположенного на сайте fstec.ru.

– База данных угроз безопасности информации Common Vulnerabilities and Exposures (CVE) http://cve.mitre.org/data/downloads/index.html .

– IBM X-Force Exchange – платформа для обмена информацией об угрозах *https://exchange.xforce.ibmcloud.com/* .

– Центр реагирования на компьютерные инциденты российский CERT/CC *www.cert.ru* .

– База данных уязвимостей SecurityFocus www.securityfocus.com .

1.6.1 Анализ угроз и оценка рисков безопасности информации

Анализ угроз включает идентификацию угроз безопасности и их оценку (более подробнее см.[4, 31]).

Идентификация угроз безопасности включает [4]:

– выявление угроз, актуальных для данной КС на всех этапах её жизненного цикла на основе таксономических классификационных перечней угроз (каталогов угроз), закрепляемых в нормативно-методических документах;

– определение спецификации: источники/природа; активы/функции, подвергаемые воздействию; методы/способы/особенности реализации; используемые уязвимости.

Анализ угроз является одной из составляющих оценки *рисков безопасности информации*.

Под риском информационной безопасности (information security risk) согласно национальному стандарту Российской Федерации Гост Р ИСО/МЭК 27005-2010: «Информационная технология. Методы и средства обеспечения безопасности. Менеджмент риска информационной безопасности, понимают «возможность того, что данная угроза сможет воспользоваться уязвимостью актива или группы активов и тем самым нанесет ущерб организации» [5].

Существуют различные методологии оценки рисков. Примеры таких методологий даны в международном отчете ИСО/МЭК ТО 13335-3:1998 "Руководство по управлению безопасностью информационных технологий. Часть 3. Методы управления безопасностью информационных технологий" [12]. Различают качественную и количественную оценку риска (risk estimation): «Процесс присвоения значений вероятности и последствий риска [ИСО/МЭК Руководство 73:2002]» [5].

Под простой количественной оценкой риска (R) понимают потенциальные потери от угроз защищенности, определяемые как мультипликативная сверстка стоимости информации и вероятности взлома (удачной попытки несанкционированного доступа к информации) [41]:

$$R(p) = C_{инф} \bullet p_{взл}.$$

Или как потери в единицу времени:

$$R(\lambda) = C_{инф} \bullet \lambda_{взл}$$

где $\lambda_{взл}$ – интенсивность потока взломов.

Риск представляет собой концепцию, связанную с обеспечением БИ. Риски и управление рисками являются неотъемлемой частью обеспечения ИБ. Целью ИБ является управление рисками, связанными с генерированием и хранением информации, будь то на компьютере, на бумаге или в любом другом формате.

Известный американский криптограф, ведущий специалист по КБ Брюс Шнайер отмечает, что истинная безопасность является математической, основанной на сравнении вероятности риска с эффективностью контрмер.

Управление безопасностью сосредоточено на управлении и минимизации риска. Целью ИБ является правильная оценка риска для обеспечения адекватной безопасности по разумной цене, при этом *остаточный риск* после принятия мер не должен превышать приемлемый уровень (рисунок 1.3).

Нет такой вещи, как совершенная безопасность, а сила контрмеры должна быть выбрана надлежащим образом для закрытия уязвимости актива. База данных описаний продуктов в электронной торговле продуктами не может быть особенно конфиденциальной и может быть защищена только длинными сложными паролями [44].

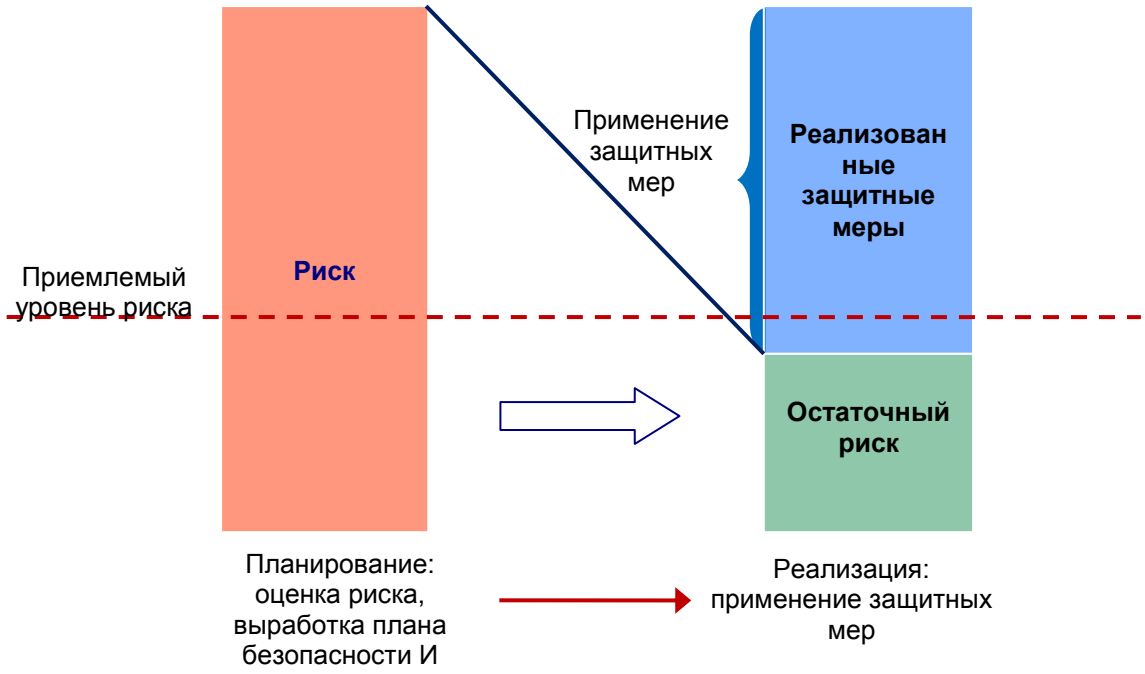


Рисунок 1.3 – Взаимосвязь защитных мер и риска

Их база данных информации о клиентах, содержащая информацию о кредитной карте, гораздо более уязвима и может потребовать, как длинного, сложного пароля, так и отпечатка пальца для разрешения доступа.

Решение о том, как сильно рискует организация, – очень сложный процесс, а классический анализ рисков мало помогает.

Несколько факторов способствуют тому, что классический анализ рисков не работает.

Во-первых, обычно существует взаимосвязь между мерами защиты и защищенными ресурсами. Например, один брандмауэр может защищать сервер и несколько рабочих станций. Тот же сервер, вероятно, защищен не только брандмауэром, но и антивирусным программным обеспечением, системой обнаружения вторжений и другими мерами безопасности. Таким образом, определение того, какая часть стоимости защиты может быть отнесена к одному активу, затруднительна, если не невозможна.

Другая, и, возможно, более сложная задача состоит в том, что вероятность возникновения определенного типа событий в значительной степени непознаваема. Даже зная, какие типы атак, с которыми организация столкнулась в прошлом году, не предсказывают, что произойдет в следующем году. Эти и другие факторы делают практически невозможным даже определить, действительно ли данная инвестиция «оправдывает себя» с точки зрения отдачи от инвестиций.

Решение заключается в том, чтобы вместо того, чтобы пытаться использовать жесткие цифры, фирма может хорошо обслуживаться путем определения приоритетности активов на основе их критичности и уязвимости информации, содержащейся в системах. После этого улучшения безопасности могут быть приоритетными, и в конкретном году наиболее важные оставшиеся активы могут быть защищены в рамках бюджетов этого года.

Например, по мере того, как операционные системы достигают конца жизни, как это недавно произошло с Windows XP, и вскоре с Windows Server 2003 угроза нападений на программное обеспечение, которое больше не получает исправлений, значительно возрастает. Поэтому приоритет должен быть уделен замене этих ресурсов, а затем привлечению внимания к следующим важнейшим активам.

Современные методы оценки рисков развиты от качественного до сочетания количественного и качественного и развиваются в направлении использования интеллектуальных методов, что позволяет оценивать риски сложных систем в условиях неопределенности [45].

1.7 Политика безопасности

Одним из лучших способов защиты от атак является в первую очередь их предотвращение. Предоставляя строгие средства определения того, кто имеет доступ к различным частям информации, часто можно предотвратить атаки на конфиденциальность, целостность и анонимность. В этом подразделе будут рассмотрены термины в области управления доступом, – виды политик разграничения доступа. Управление доступом основано на понятии политики (формализованных правил, алгоритмов, механизмов).

Под *политикой безопасности* КС понимают совокупность свойств, принципов, правил и механизмов обеспечения защищенности информации в КС в заданном пространстве угроз.

Модель угроз – составная часть политики безопасности, описывающая возможные угрозы, возможности нарушителя, может содержать оценку угроз.

Модель безопасности – формальное (математическое, алгоритмическое, схемотехническое и т.п.) выражение политики безопасности.

Модели безопасности обеспечивают системотехнический подход в процессах разработки и исследования защищенных КС, включающий решение следующих задач [4]:

- обоснованного выбора базовых принципов архитектуры, определяющих механизмы реализации средств защиты информации
- доказательство защищенности КС;
- составления формальной спецификации политики безопасности КС.

Модель безопасности состоит из:

- модели КС;
- критериев, принципов или целевых функций защищенности и угроз;
- политик безопасного функционирования КС.

Большинство моделей КС относится к классу моделей конечных состояний. Изложение сути субъектно-объектной модели КС в механизмах и процессах

коллективного доступа к информационным ресурсам, понятия монитора безопасности и требования к нему, также понятия и характеристик основных типов политик безопасности можно посмотреть в [4, 41].

Разграничение доступа к информации КС включает:

− разделение всей информации КС на объекты с организованным доступом субъектов только к той части информации, которая необходима им по определенным, тем или иным, соображениям;

− организация данных и информационных процессов, таким образом, чтобы обеспечить безопасность КС.

Доступ к информации осуществляется методами доступа (т.е. различные действиями субъектов над объектами КС, основными являются чтение и запись), вызывающими одно-двунаправленные информационные потоки.

Права доступа – методы доступа, назначенные субъектам.

Политика разграничения доступа подразумевает политику назначения прав доступа субъектов КС объектам и политику осуществления доступов.

Виды политик разграничения доступа отображены на рисунке 1.4.

Все модели предполагают наличие *диспетчеров данных*, владельцев данных или системных администраторов, которые определяют спецификации контроля доступа.

Цель состоит в том, что эти люди должны ограничивать доступ к тому, что имеет необходимость доступа и / или изменения соответствующей информации – ресурсам. То есть, они должны применять принцип наименьших привилегий.

1.8 Меры и средства защиты информационных процессов в компьютерных системах

Защита информационных процессов в КС осуществляется путем принятия совокупности правовых, организационных и технических *мер*, направленных на [26]:

- блокирование (нейтрализацию) угроз БИ;

– на локализацию и минимизацию последствий от возможной реализации угроз БИ;

– восстановление штатного режима функционирования КС в случае реализации угроз БИ [36];

– обеспечение ЗИ от неправомерного доступа, уничтожения, модифицирования, блокирования, копирования, предоставления, распространения, а также от иных неправомерных действий в отношении такой информации;

– соблюдение конфиденциальности информации ограниченного доступа;

– реализацию права на доступ к информации [26].

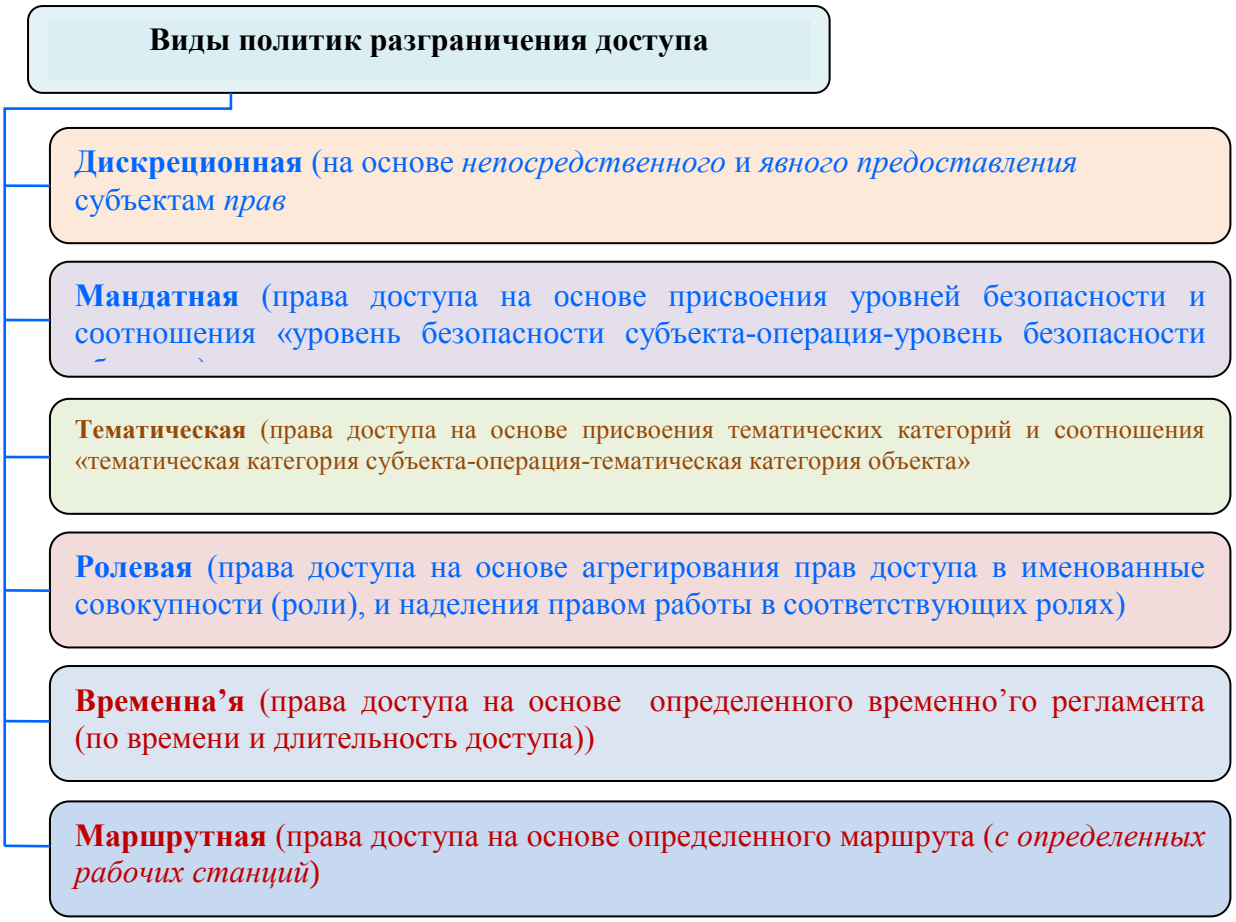


Рисунок 1.4 – Виды политик разграничения доступа

Выбор мер защиты информации выполняется согласно целесообразности, экономичности, комплексности, системности, а также в соответствии требованиям законодательства Российской Федерации и иных нормативных документов.

В частности, выбор мер защиты информации в государственной информационной системе осуществляется «в зависимости от класса защищенности информационной системы, угроз безопасности информации, структурно-функциональных характеристик информационной системы, применяемых информационных технологий и особенностей функционирования информационной системы» [34].

В информационной системе подлежат реализации следующие меры защиты информации:

- идентификация и аутентификация субъектов доступа и объектов доступа;
- управление доступом субъектов доступа к объектам доступа;
- ограничение программной среды;
- защита машинных носителей информации;
- регистрация событий безопасности;
- антивирусная защита;
- обнаружение (предотвращение) вторжений;
- контроль (анализ) защищенности информации;
- обеспечение целостности информационной системы и информации;
- обеспечение доступности информации;
- защита среды виртуализации;
- защита технических средств;
- защита информационной системы, ее средств и систем связи и передачи данных.

Выбор мер определяется также моделью угроз БИ: меры защиты информации должны обеспечивать блокирование одной или нескольких угроз безопасности информации, включенных в модель угроз безопасности информации. Общий порядок действий по выбору и мер защиты информации для их реализации в информационной системе представлен в методическом документе «Меры защиты информации в государственных информационных системах» [34].

Основой для выбора мер и средств также является оценка рисков (рисунок 1.3) и соблюдение принципов БИ.

Выделен ряд *общих принципов* создания и эксплуатации защищенных компьютерных систем, такие как принцип разумной достаточности, целенаправленности, системности, комплексности, непрерывности, управляемости, сочетания унификации и оригинальности, подробное разъяснение принципов, а также классификацию доступных методов и механизмов обеспечения КБ см. в [4].

Наиболее полным документом по классификации принципов БИ можно считать «Engineering Principles for Information Technology Security (ABaseline for Achieving Security), Revision A», публикуемый Национальным институтом стандартов и технологий (NIST). В этом документе представлены общие принципы, применимые ко всем ИТ-системам, содержит 33 принципа безопасности ИТ, сгруппированных в следующие 6 категорий: «Базовая безопасность», «Основанные на рисках», «Простые в использовании», «Повышение устойчивости», «Устранение уязвимостей» и «Разумный проектирование сети». Предполагается, что применение этих общих принципов к конкретным технологическим областям будет достигнуто путем использования этого документа при разработке более подробных рекомендаций. Более того, из-за постоянно меняющейся информационной среды информационной системы выявленные принципы не считаются статическим, всеохватывающим списком. Вместо этого этот документ представляет собой попытку представить логически обоснованные принципы безопасности, которые могут использоваться в современных КС. По мере совершенствования технологий и совершенствования техники безопасности потребуются дополнения, исключения и уточнение этих принципов безопасности [47].

Одним из главных направлений работ в области создания методов и средств обеспечения БИ является задача разработки целостной теории защиты информации. Основные понятия теории защиты, задачи, состав и общее содержание, научно-методологическая основа (общеметодологические принципы формирования теории, методы решения задач и методологический базис) ЗИ изложены в работах Герасименко В.А., Малюк А.А., в том числе в работе [18] и др.

Контрольные вопросы

1 Как соотносятся понятия информационная безопасность и компьютерная безопасность?

2 Что понимается под информационной безопасностью согласно документу «ДОКТРИНА информационной безопасности Российской Федерации»?

3 Что относится к объектам информационной безопасности?

4 Что представляют собой информационные процессы КС?

5 Какие компоненты включает КС как объект защиты?

6 По каким признакам можно классифицировать КС?

7 Какую информацию нужно защищать?

8 Какая информация относится к информации ограниченного доступа согласно ФЗ №149, 2006г.?

9 Классически информационная безопасность была определена в Что понимают под акронимом C.I.A. при определении информационной безопасности?

10 Что понимается под *конфиденциальностью* информации?

11 Какие существуют инструменты для защиты конфиденциальной информации?

12 Что понимается под целостностью информации (данных)?

13 Какие существуют инструменты специально предназначенные для обеспечения целостности?

14 Что понимается под доступностью информации (данных)?

15 Какие существуют инструменты для обеспечения доступности?

16 Что означает трехбуквенный акроним A.A.A. (Assurance, Authenticity, and Anonymity)?

17 В какие категории сгруппированы принципы безопасности ИТ в документе, публикуемом Национальным институтом стандартов и технологий (NIST)?

2 Основные понятия и концепции криптографии

Существует множество аспектов компьютерной и информационной безопасности. Шифрование, процесс скремблирования сообщения или другой информации, в такую форму, чтобы оно не могло быть легко прочитано посторонними, является одной из наиболее важных частей в области безопасности. Если в организации есть лучший брандмауэр, очень жесткие политики безопасности, защищенные операционные системы, антивирусные сканеры, программное обеспечение для обнаружения вторжений, антишпионские программы и все другие механизмы безопасности КС, но данные отправляются в открытом, обычном тексте, тогда она просто небезопасна.

Криптографические методы можно использовать для достижения широкого круга целей безопасности, в том числе и тех, которые вначале могут казаться невозможными.

В этом разделе рассматривается криптография - искусство написания или расшифровки секретного кода. Раздел предназначен для того, чтобы дать общее представление о том, что такое шифрование, как оно работает, и достаточно информации для принятия разумного решения о том, какие типы шифрования должны быть использованы в организации.

Рассматриваются фундаментальные концепции и некоторые технические детали. Например, фактический процесс некоторых криптографических алгоритмов. Но основное внимание уделяется широкому пониманию соответствующих концепций.

Основная цель криптографии – не скрытие существования сообщения, а скорее скрытие его смысла путем процесса, известного как шифрование. Чтобы сделать сообщение непонятным, оно скремблируется в соответствии с определенным алгоритмом, который заранее согласован между отправителем и предполагаемым получателем, таким образом, что получатель может изменить протокол скремблирования и сделать сообщение понятным. Это изменение скремблирования называется расшифровкой [44].

Криптология – наука, предметом которой являются математические основания криптографии и криптоанализа.

Криптоанализ – наука о раскрытии шифра без знания ключа (проверка устойчивости шифра).

Криптографические алгоритмы позволяют обеспечить:

- конфиденциальность передаваемых данных;
- аутентичность (подлинность, целостность) данных;
- подтверждение подлинности пользователя.

2.1 Шифрование

Идея шифрования, вероятно, такая же старая, как и письменная связь. Основная концепция на самом деле довольно проста. Сообщения должны быть изменены таким образом, чтобы не могли быть легко прочитаны врагом, но могли быть легко восстановлены предполагаемым получателем.

Простой сценарий шифрования описывается как средство, позволяющее двум сторонам, например, пользователю А и пользователю В, устанавливать конфиденциальную связь по небезопасному каналу, в котором возможно подслушивание.

Шифрование также имеет другие применения и приложения, но понимание простого сценария дает основу, на которой можно затем строить расширения.

Предположим, что у пользователя *А* есть конфиденциальное сообщение, М, для пользователя Б, которое он хочет переслать ему по открытым каналам.

Сообщение *M* называется *открытым текстом*, и оно не должно передаваться в открытой форме, поскольку оно может быть прочитанным другими сторонами во время передачи. Вместо этого пользователь А преобразует открытый текст *M* в зашифрованную форму с использованием алгоритма шифрования *E* , который дает на выходе *зашифрованный текст C* для *M* .

Этот процесс шифрования обозначается *C* = *E* (*M*).

Криптограмма *C* будет фактически передана пользователю Б. Пользователь Б получив *C* , применит алгоритм расшифровки *D* для восстановления исходного открытого текста M из зашифрованного текста *C* .

Этот процесс дешифрования обозначается $M = D(C)$.

Алгоритмы шифрования и дешифрования выбираются так, что для любого субъекта, кроме пользователя *А* и пользователя *Б*, невозможно определить открытый текст *M* из зашифрованного текста *C* . Таким образом, шифрованный текст *C* может передаваться по небезопасному каналу, который может быть перехвачен противником.

Для понимания последующего материала, следует начать с некоторых базовых понятий:

– *Шифр* (стандарт ГОСТ ГОСТ Р 34.12-2015) – «криптографический метод, используемый для обеспечения конфиденциальности данных, включающий алгоритм зашифрования и алгоритм расшифрования» [8].

– *Ключ* – конкретные секретные параметры криптографического алгоритма, обеспечивающие выбор только одного варианта из всех возможных для данного алгоритма; биты, которые объединяются с простым текстом для его шифрования. В некоторых случаях это случайные числа; в других случаях это результат некоторой математической операции.

– *Открытый текст*: исходный незашифрованный текст.

– *Криптограмма*: зашифрованный текст.

– *Алгоритм шифрования*: математический процесс для шифрования.

– *Пространство ключей* – множество всех возможных ключей криптоалгоритма;

– *Лавинный эффект* – малое различие между двумя наборами входных данных превращается в большое на выходе.

Современные криптографические алгоритмы можно разделить на три группы по количеству используемых ключей следующим образом (рисунок 2.1):

Рисунок 2.1 – Классификация криптографических алгоритмов

Выделяют два основных типа алгоритмов шифрования: симметричный и асимметричный. *Симметричный* (одноключевой) алгоритм использует один и тот же ключ (секретный) и для зашифрования и для дешифрования сообщения. *Асимметричный* (двухключевой) алгоритм для зашифрования использует один ключ (общедоступный), а для дешифрования сообщения использует другой ключ (закрытый).

Также выделяют алгоритмы шифрования:

– алгоритм замены (подстановки) – символы шифруемого текста заменяются символами того же или другого алфавита в соответствии с заранее обусловленной схемой замены;

– алгоритм перестановки – символы шифруемого текста переставляются по определенному правилу в пределах некоторого блока этого текста;

– гаммирование – символы шифруемого текста складываются с символами некоторой случайной последовательности;

– аналитическое преобразование – преобразование шифруемого текста по некоторому аналитическому правилу (формуле) [17].

Процессы шифрования осуществляются посредством определенной криптосистемы.

Криптографическая система включает алгоритм шифра и схему управления ключами.

Примеры алгоритмов – описания AES, Магма, Кузнечик.

2.2 Симметричные криптосистемы

Симметричные криптосистемы (с секретным ключом – secret key systems). Шифрование и расшифрование производится одним и тем же секретным ключом. Секретность ключа является постулатом. Основная проблема при применении симметричных криптосистем для связи заключается в сложности передачи обоим сторонам секретного ключа. Симметричные алгоритмы делят на блочные и поточные.

В *блочных шифрах* результат зашифрования очередного блока зависит только от него самого и не зависит от других блоков шифруемого массива данных:

$$C_i = E(P_i).$$

В результате зашифрования двух одинаковых блоков открытого текста одним алгоритмом получаются идентичные блоки шифротекста:

$$P_i = P_j \Rightarrow E(P_i) = E(P_j).$$

В *поточных* или *потоковых шифрах* результат зашифрования очередного блока зависит от него самого и, в общем случае, от всех предыдущих блоков массива данных:

$$C_i = E(P_1, P_2, \ldots, P_i).$$

На основе ключа вырабатывается некая последовательность – так называемая *выходная гамма*, которая затем накладывается на текст сообщения.

2.2.1 Требования к симметричным шифрам

Согласно *принципу Кирхгофа* шифр представляется в виде параметризованного алгоритма, включающего *процедурную часть* и *параметры* – различные элементы данных, используемые в преобразованиях.

Раскрытие только процедурной части не должно приводить к увеличению вероятности успешного дешифрования сообщения злоумышленником выше допустимого предела [15, 30].

Поэтому процедурную часть не скрывают, секретным является только *ключ* шифра.

Следствия:

– разглашение конкретного шифра (алгоритма и ключа) не приводит к необходимости полной замены реализации всего алгоритма, достаточно заменить только скомпрометированный ключ;

– ключи можно отчуждать от остальных компонентов системы шифрования – хранить отдельно от реализации алгоритма в более надежном месте и загружать их в шифрователь только по мере необходимости и только на время выполнения шифрования.

Условия стойкости блочного шифра (по К. Шеннону):

Рассеивание (diffusion) – один бит исходного текста должен влиять на несколько битов шифротекста, оптимально – на все биты в пределах одного блока. Аналогично и для зависимости шифротекста от ключа – один бит ключа должен влиять на несколько битов шифротекста. Это помогает создать лавинный эффект, скрыть статистические свойства открытого текста [30, 39];

Перемешивание (confusion) – использование преобразований, затрудняющих получение зависимостей между шифротектстом и открытым текстом. Если шифр достаточно хорошо «перемешивает» биты исходного текста, то соответствующий шифротекст не содержит никаких статистических, и, тем более, функциональных закономерностей [30, 38].

Если шифр обладает обоими указанными свойствами, то любые изменения в блоке открытых данных приведут к тому, что все биты в зашифрованном блоке данных с вероятностью ½ независимо друг от друга так же поменяют свои значения.

Такой шифр невозможно вскрыть способом, менее затратным с точки зрения количества необходимых операций, чем полный перебор по множеству возможных значений ключа.

Шеннон предложил идею построения криптографически стойкой системы путем последовательного применения относительно простых *шифрующих* преобразований (идея многократного шифрования), т.е. получение сложного преобразования на основе *композиции* простых преобразований: $F=F_1oF_2oF_3oF_4o..oF_n$, где F - преобразование шифра, F_i – простое преобразование, называемое также i-ым *циклом шифрования*. Например, стандарт DES состоит из 16 циклов. Если же используется одно и то же преобразование, то такой композиционный шифр называют *итерационным шифром*.

Простое шифрующие преобразование – преобразование, которое реализуется аппаратно относительно несложной логической схемой или программно несколькими компьютерными командами.

К основным шифрующим преобразованиям относятся:

– перестановка (permutation) – перестановка структурных элементов шифруемого блока данных (битов, символов, цифр);

– замена, подстановка (substitution) – замена группы элементов шифруемого блока на другую группу по индексной таблице;

– функциональное преобразование (function) – различные сдвиги, логические и арифметические операций.

Наибольшую популярность имеют шифры, основанные на *сети Фейстеля* (Файстеля - Feistel), в которых:

1 Входной блок p для каждого преобразования разбивается на два подблока: $p=(l,r)$, где l – левая часть, а r – правая часть;

2 Используется преобразование вида $F_i(l,r)=(r,l\otimes f_i(r,K_i))$, где f_i – зависящая от ключа K_i функция, а $\otimes$ – операция XOR или некая другая.

Функция f_i называется *раундовой (цикловой) функцией*, а ключ K_i, используемый для получения функции f_i называется *раундовым (цикловым) ключом*.

Это преобразование прокручивается несколько раз (несколько циклов) и выходом шифра является получившаяся в конце пара (l,r) .

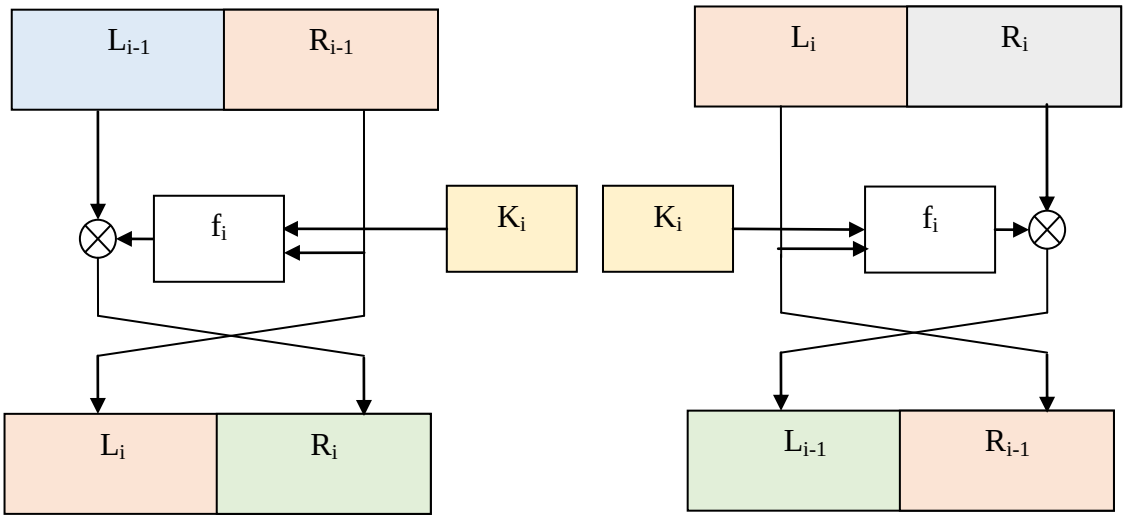
Для любой (необязательно обратимой) функции f_i расшифрование осуществляется путем выполнения тех же процедур преобразования, но с использованием *подключей* K_i в обратном порядке.

Подключи K_i генерируются из ключа K специальными алгоритмами, разрабатываемыми вместе с шифром.

Размер левой и правой частей блока может изменяться от раунда к раунду, но обычно эти величины постоянны. Если они равны друг другу, то такая схема называется *сбалансированной*, а если нет – то *несбалансированной* сетью Фейстеля.

Обычно «°» – операция побитового исключающего ИЛИ, если используется другая подходящая бинарная операция, то сеть Фейстеля называется *обобщенной*.

Графически сеть Фейстеля выглядит следующим образом (рисунок 2.2):



$L_i=R_{i-1}$
$R_i=L_{i-1}\otimes f_i(R_{i-1},K_i))$, i=1,2..16

$R_{i-1}=L_i$
$L_{i-1}=R_i\otimes f_i(L_i,K_i))$, i=16,15..1

Рисунок 2.2 – Пример одного цикла преобразований сети Фейстеля

В качестве функции f_i выступает некая комбинация перестановок, подстановок, сдвигов, добавлений ключа и прочих преобразований. Так, при использовании подстановок информация проходит через специальные блоки, называемые *S-блоками (S-боксами, S-boxes)*, в которых значение группы битов

заменяется на другое значение. По такому принципу (с небольшими отличиями) построены многие алгоритмы: DES, Магма и т.п.

В других алгоритмах используются несколько иные принципы. Так, например, алгоритмы, построенные по *SP-принципу* (*SP-сети*) осуществляют преобразование, пропуская весь входной блок, а не его половину, через последовательность подстановок (*Substitutions*) и перестановок (*Permutations*). Отсюда и название – *SP-сети*, т.е. сети "подстановок-перестановок". Примером такого алгоритма является разработка Rijndael (AES), Кузнечик. Размеры блоков в каждом алгоритме свои.

Получение цикловых ключей.

Ключ имеет фиксированную длину. Однако при прокрутке хотя бы 8 циклов шифрования с размером блока, скажем, 128 бит даже при простом прибавлении посредством XOR потребуется 8*128=1024 бита ключа, поскольку нельзя добавлять в каждом цикле одно и то же значение, так как это ослабляет шифр. Поэтому для получения последовательности ключевых бит разрабатывают специальный *алгоритм выработки раундовых ключей (ключевое расписание – key schedule)*. В результате работы этого алгоритма из исходных бит ключа шифрования получается массив бит определенной длины, из которого по определенным правилам составляются цикловые ключи [17].

2.2.2 Российский стандарт симметричных блочных методов

Российский новый стандарт блочного шифрования – ГОСТ Р 34.12-2015 «Информационная технология. Криптографическая защита информации. Блочные шифры» разработан Центром защиты информации и специальной связи ФСБ России с участием ОАО «Информационные технологии и коммуникационные системы» (ОАО «ИнфоТеКС»). Внесен Техническим комитетом по стандартизации ТК 26 «Криптографическая защита информации». Шифр утвержден приказом от 19 июня 2015 года № 749-ст. Стандарт вступил в действие с 1 января 2016 года [8].

Этот стандарт помимо старого ГОСТ 28147-89, принятого в 1989 году который теперь называется «Магма» и имеет фиксированный набор подстановок, содержит описание блочного шифра «Кузнечик».

Магма

Магма – симметричный блочный алгоритм шифрования с 256-битным ключом, шифрующий данные по 64 битным блокам. Аналогично шифру DES основан на сети Фейстеля и нелинейных преобразований в форме S-блоков (рисунок 2.3).

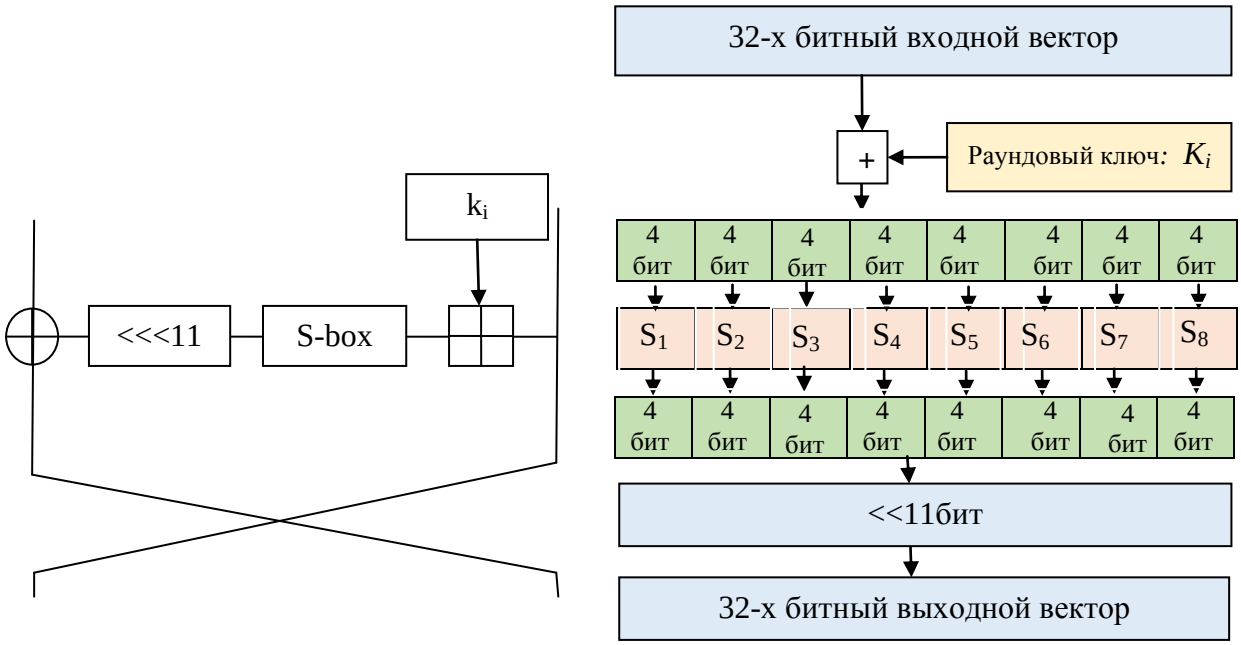


Рисунок 2.3 – Пример одного цикла преобразований в Магма

На рисунке 2.4 приведена реализация метода Transform класса GostECB, выполняющего шифрование блока data по алгоритму Магма (https://github.com/yaruson/GostPlugin).

Реализация раундовой функции F() использует тип uint для простоты выполнения сложения по модулю ключа и 32-битной части исходного блока и циклического сдвига на 11 бит (рисунок 2.5).

Метод подстановки по S-блокам работает с 4-битными частями 32-битного подблока, которые отделяют побитовым сдвигом и дальнейшим умножением на 0x0f (рисунок 2.6).

```
public static byte[] Transform(byte[] data, byte[] key, byte[][] sBox, bool encrypt)
{   // разделение на левый и правый подблоки и преобразование из байт в биты
    var a = BitConverter.ToUInt32(data, 0);
    var b = BitConverter.ToUInt32(data, 4);
    var subKeys = GetSubKeys(key);
    var result = new byte[8];
    for (int i = 0; i < 32; i++) {  // выполнить 32 раунда
        var keyIndex = GetKeyIndex(i, encrypt);
        var subKey = subKeys[keyIndex];
       var fValue = F(a, subKey, sBox);
        var round = b ^ fValue;
        if (i < 31)  { //если не последний раунд
            b = a;
            a = round;
        }
        else { // для последнего раунда
            b = round;
        }
    }
// разделение на левый и правый подблоки и преобразование из байт в биты
    Array.Copy(BitConverter.GetBytes(a), 0, result, 0, 4);
    Array.Copy(BitConverter.GetBytes(b), 0, result, 4, 4);
    return result;
}
```

Рисунок 2.4 – Метод класса для шифрования Магма

```
private static uint F(uint block, uint subKey, byte[][] sBox)
{
    block = (block + subKey) % uint.MaxValue;
    block = Substitute(block, sBox);
    block = (block << 11) | (block >> 21);
    return block;
}
```

Рисунок 2.5 – Раундовая функция F

```
private static uint Substitute(uint value, byte[][] sBox) {
    byte index, sBlock;
    uint result = 0;
    for (int i = 0; i < 8; i++) {
        index = (byte)(value >> (4 * i) & 0x0f);
        sBlock = sBox[i][index];
        result |= (uint)sBlock << (4 * i);
    }
    return result;
}
```

Рисунок 2.6 – Подстановка по S-блокам в Магма

Шифрование от расшифровывания в режиме простой замены отличается порядком использования ключей. На самом деле, применительно к режиму

гаммирования с обратной связью нам не надо ничего расшифровывать, однако для полноты реализации можно предусмотреть и эту возможность (рисунок 2.7).

```
private static int GetKeyIndex(int i, bool encrypt)
{
    return encrypt ? (i < 24) ? i % 8 : 7 - (i % 8)
                   : (i < 8) ? i % 8 : 7 - (i % 8);
}
```

Рисунок 2.7 – Метод определения порядка использования ключей

Один из режимов работы Магма, режим гаммирования с обратной связью, является потоковым режимом блочного шифра.

Описание алгоритма в режиме гаммирования

1 Исходное сообщение разбивается на блоки по 64 бита.

2 На каждый блок XOR'ом «накладывается» гамма, тоже длиной 64 бита.

3 Гамма формируется шифрованием 64-битного блока «состояния» с помощью ключа в режиме простой замены.

– В момент начала шифрования сообщения блок принимается равным синхропосылке или вектору инициализации.

– В следующей итерации вместо синхропосылки используется зашифрованный блок текста из предыдущей.

Приведённая последовательность действий справедлива как для шифрования, так и расшифрования. Разница в том, откуда берётся зашифрованный блок текста для обработки следующего блока.

Ниже приведён фрагмент кода класса, выполняющего поблочное преобразование Магма в режиме гаммирования. При создании экземпляра класса в _state записывается значение синхропосылки, затем в него заносится очередной блок зашифрованных данных (рисунок 2.8).

Исходные коды реализации алгоритма доступны на github [50].

Блочный шифр «Кузнечик» (Kuznechik, Kuznyechik) – симметричный алгоритм блочного шифрования с размером блока 128 бит и длиной ключа 256 бит [8].

```
public int TransformBlock (byte[] inputBuffer, int inputOffset, int inputCount, byte[]
outputBuffer, int outputOffset)
{
  byte[] dataBlock = new byte[inputCount];
  byte[] gamma = new byte[GostECB.BlockSize];
  byte[] result = new byte[inputCount];
  Array.Copy(inputBuffer, inputOffset, dataBlock, 0, inputCount);

  gamma = GostECB.Transform(_state, _key, GostECB.SBox_CryptoPro_A, true);
  result = XOr(dataBlock, gamma);

  Array.Copy(result, 0, outputBuffer, outputOffset, inputCount);
  Array.Copy(_encrypt ? result : dataBlock, _state, inputCount);

  return inputCount;
}
```

Рисунок 2.8 – Фрагмент кода шифрования Магма

В отличие от Магмы, Кузнечик представляет собой не сеть Фейстеля, а SP-сеть, схожую с шифром AES, но с рядом отличий:

– линейное преобразование может быть реализовано с помощью регистра сдвига;

– ключевая развертка реализована с помощью сети Фейстеля, в которой в качестве функции используется раундовое преобразование исходного алгоритма.

Преобразования шифра Кузнечик

Шифр принадлежит к классу *LSX*-шифров. Шифрование основано на последовательном применении десяти однотипных раундов, каждый из которых содержит три преобразования: X (смешивание с цикловым ключом) – сложение с раундовым ключом k , S (подстановка) – преобразование блоком подстановок S-box и линейное преобразование L (рисунок 2.9). Преобразование X: 128-битный входной вектор очередного раунда $a \in V_{128}$ складывается побитно (операция $\oplus$) с раундовым ключом $k \in V_{128}$:

$$X[k]: V_{128} \to V_{128}\,; X[k](a) = k \oplus a, \text{где } k, a \in V_{128}$$

Пример преобразования X на C# приведен на рисунке 2.10.

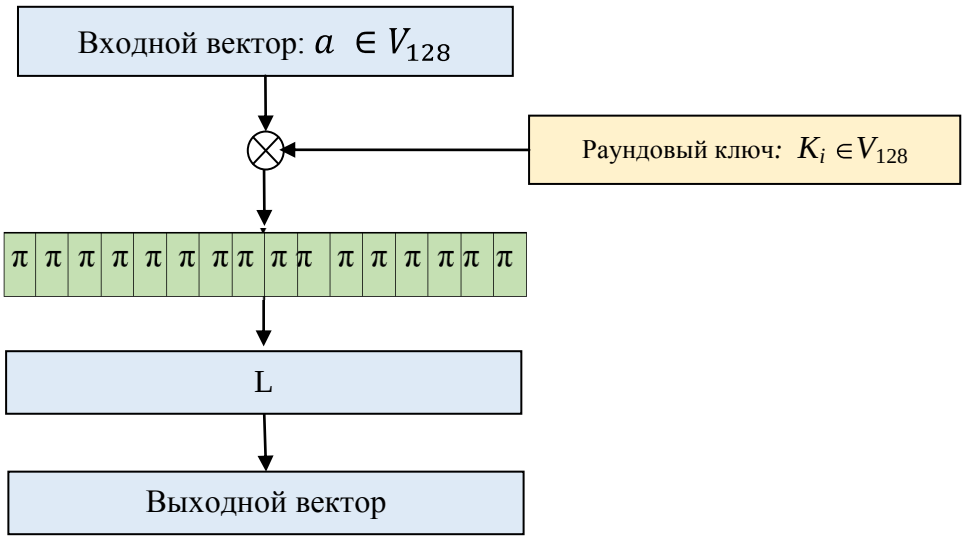


Рисунок 2.9 – Раундовое преобразование

```
private void X (ref byte[] result, ref byte[] data) {
            for (int i = 0; i < result.Length; i++) {
                result[i] ^= data[i];
            }
        }
```

Рисунок 2.10 – Сложение блока данных с раундовым ключом на C#

Нелинейное преобразование S представляет собой применение к каждому 8-битному подвектору $a_i \in V_8$ 128-битного входного вектора $a \in V_{128}$ фиксированной подстановки π:

$$S: V_{128} \to V_{128}\,; S(a) = S(a_{15}||\dots||a_0) = \pi(a_{15})||\cdots||\pi(a_0),$$

$$\text{где } a = a_{15}||\dots||a_0 \in V_{128};\ a_i \in V_8, i = 1, \cdots 15$$

Пример преобразования S на C# приведен на рисунке 2.110.

```
private void S (ref byte[] data) {
            for (int i = 0; i < data.Length; i++) {
                data[i] = _pi[data[i]];
            }
        }
```

Рисунок 2.11 – Преобразование S на C#

Линейное преобразование может быть реализовано матрицей, или с помощью РСЛОС – линейного регистра сдвига с обратной связью, который движется 16 раз:

$R: V_{128} \to V_{128}$; $R(a) = R(a_{15}|| \dots ||a_0) = l(a_{15}, \dots, a_0)||(a_{15})|| \cdots ||(a_1)$,

где $a = a_{15}|| \dots ||a_0 \in V_{128}$; $a_i \in V_8, i = 0,1, \cdots, 15$;

$L: V_{128} \to V_{128}$; $L(a) = R^{16}(a)$, где $a \in V_{128}$

Сам регистр реализуется над полем Галуа по модулю неприводимого многочлена степени 8: $p(x) = x^8 + x^7 + x^6 + x + 1$ (рисунок 2.12).

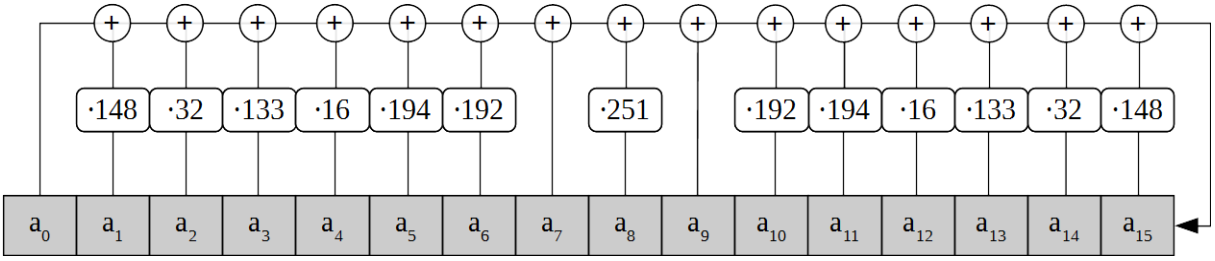


Рисунок 2.12 – Регистр сдвига с обратной связью

Пример преобразования L на C# приведен на рисунке 2.13.

```
private void R (ref byte[] data) {
            byte z = l(ref data);
            for (int i = 15; i > 0; i--) {
                data[i] = data[i - 1];
            }
            data[0] = z;
        }
private void L (ref byte[] data) {
            for (int i = 0; i < 16; i++) {
                R(ref data);
            }
        }
```

Рисунок 2.13 – Преобразование S на C#

Пример фрагмента программной реализации алгоритма шифрования Кузнечик приведен на рисунке 2.14. Программную реализацию на языке C# можно посмотреть в [50].

Раундовые ключи получают из главного ключа K. Для получения ключей K_1 K_2 ключ K разбивают пополам. Далее для выработки следующей пары раундовых ключей используется 8 итераций сети Фейстеля, где, в свою очередь, в качестве раундовых ключей используется счетчиковая последовательность, прошедшая через линейное преобразование алгоритма.

```
public byte[] Encrypt (byte[] data) {
            byte[] block = new byte[BLOCK_SIZE];
            byte[] temp = new byte[BLOCK_SIZE];
            Array.Copy(data, block, BLOCK_SIZE);

            for (int i = 0; i < 9; i++) {
                LSX(ref temp, ref _subKeys[i], ref block);
                Array.Copy(temp, block, BLOCK_SIZE);
            }
            X(ref block, ref _subKeys[9]);
            return block;
        }
```

Рисунок 2.14 – Фрагмент кода шифрования Кузнечик (+)

$F[k]: V_{128} \times V_{128} \to V_{128} \times V_{128}$; $F[k](a_1, a_0) = (LSX[k](a_1) \oplus a_0, a_1)$,

где $k, a_1, a_0 \in V_{128}$

$C_i = L\left(Vec_{128}(i)\right), \quad i = 1,2, \dots ,32$

$(K_{2i+1}, K_{2i+2}) = F\left[C_{8(i-1)+8}\right] \dots F\left[C_{8(i-1)+1}\right](K_{2i-1}, K_{2i}), i = 1,2,3,4.$

Пример выработки ключей на C# приведен на рисунке 2.15.

```
public void SetKey (byte[] key) {
            _subKeys = new byte[10][];
            for (int i = 0; i < 10; i++) {
                _subKeys[i] = new byte[SUB_LENGTH];
            }
            byte[] x = new byte[SUB_LENGTH];
            byte[] y = new byte[SUB_LENGTH];
            byte[] c = new byte[SUB_LENGTH];
            /** SubKey[1] = k[255..128]
             * SubKey[2] = k[127..0]*/
            for (int i = 0; i < SUB_LENGTH; i++) {
                _subKeys[0][i] = x[i] = key[i];
                _subKeys[1][i] = y[i] = key[i + 16];
            }
            for (int k = 1; k < 5; k++) {
                for (int j = 1; j <= 8; j++) {
                    C(ref c, 8 * (k - 1) + j);
                    F(ref c, ref x, ref y);
                }
                Array.Copy(x, _subKeys[2 * k], SUB_LENGTH);
                Array.Copy(y, _subKeys[2 * k + 1], SUB_LENGTH);
            }   }
        private void F (ref byte[] k, ref byte[] a1, ref byte[] a0) {
            byte[] temp = new byte[SUB_LENGTH];
            LSX(ref temp, ref k, ref a1);
            X(ref temp, ref a0);
            Array.Copy(a1, a0, SUB_LENGTH);
            Array.Copy(temp, a1, SUB_LENGTH);
        }
```

Рисунок 2.15 – Выработка раундовых ключей на C#

2.2.3 Режимы работы блочных шифров.

Режимы позволяют более эффективно решать различные криптографические задачи, основываясь на следующих соображениях: скрытие структуры открытого текста, затруднение манипулированием открытого текста, рандомизация ввода шифра, возможность шифрования нескольких текстов одним ключом, устойчивость к сбоям [27, 40]. Режимы не дают злоумышленникам читать поток сообщений, но при этом не обеспечивают аутентификации, следовательно, злоумышленник может изменять исходные сообщения и в результате расшифрования будут получены бессмысленные сообщения, которые однако способны повлиять на остальные части системы. Поэтому шифрование всегда нужно применять вместе с аутентификацией. Наиболее часто применяются следующие режимы:

- электронная кодовая книга – ECB (Electronic Code Book);
- сцепление блоков шифротекста – CBC (Cipher Block Chaining);
- обратная связь по шифротектсту – CFB (Cipher Feed Back);
- обратная связь по выходу – OFB (Output Feed Back).

- Важным аспектом режимов CBC, CFB, OFB является шифрование первого блока открытого текста, для которого добавляется уникальный *вектор инициализации* IV – (Init Vector). IV лучше всего генерировать путем шифрования *оказии* (nonce – number used only once). Оказия – уникальное число, присваиваемое каждому сообщению (например номер сообщения в комбинации с какой-нибудь дополнительной информацией). Оказию не обязательно держать в секрете, но использовать можно только один раз. Нарушение уникальности IV в OFB и CFBболее критично, чем в CBC.

Электронная Кодовая Книга (ECB). Режим электронной кодовой книги (ECB) предполагает использование шифра обычным образом, как он есть, без улучшения безопасности. Сообщение делится на блоки, и каждый блок зашифровывается независимо (рисунок 2.16). Проблема в том, что одинаковый открытый текст дает одинаковый шифротекст. Это позволяет злоумышленникам получить ключ.

Сцепление блоков шифротекста (Cipher-Block Chaining, CBC)

В режиме сцепления блоков шифротекста (CBC) каждый блок открытого текста перед его шифрованием складывается с предыдущим блоком шифротекста. В результате, в зашифротексте значительно больше случайности чем в режиме ECB. Это наиболее широко используемый режим. CBC хорошо защищает от известных атак с открытым текстом.

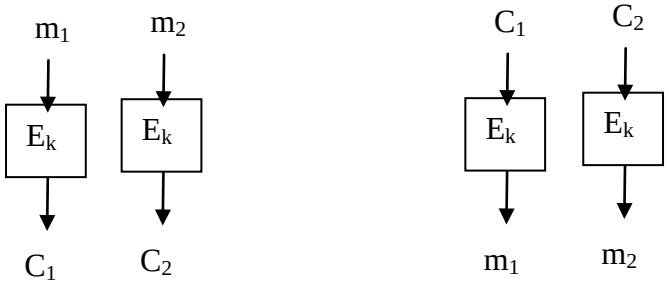


Рисунок 2.16 - Схема электронной кодовой книги

Схема использования. Исходный текст разбивается на блоки, а затем (рисунок 2.17):

1 Первый блок складывается побитно по модулю 2 (XOR) с вектором инициализации IV.

2 Полученное значение шифруется.

3 Полученный в результате блок шифротекста отправляется получателю и одновременно служит начальным вектором IV для следующего блока открытого текста.

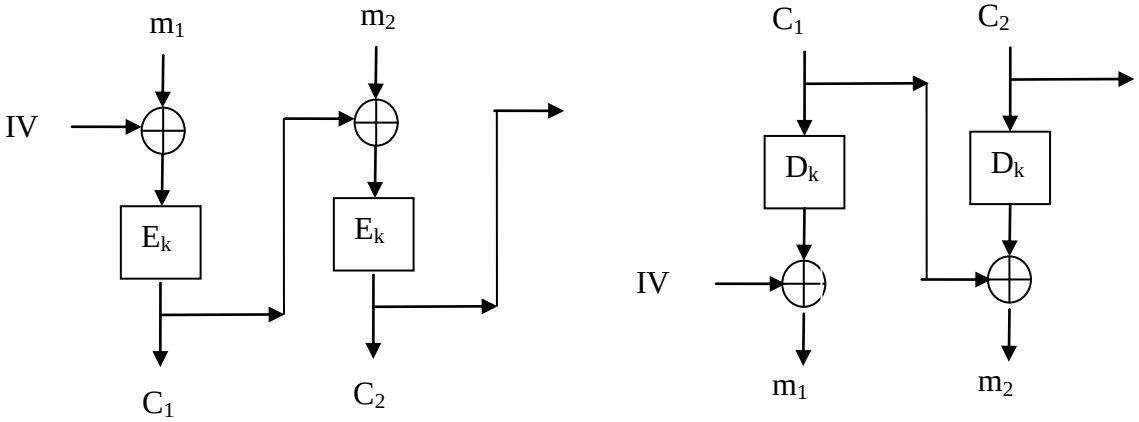


Рисунок 2.17 – Схема сцепления блоков шифротекста

Расшифрование осуществляется в обратном порядке.

Преобразование в режиме CBC можно представить в виде формул:

$C_i=E_k(M_i \oplus C_{i-1})$, $C_1=E_k(M_1 \oplus IV)$, $M_i=D_k(C_i) \oplus C_{i-1}$, $i>1$, $M_1=D_k(C_1) \oplus IV$,

где *i* – номер соответствующего блока.

Основные характеристики режима:

– Ошибка в одном бите шифротекста портит этот блок сообщения и один бит следующего блока.

– Потеря (добавление) бита полностью искажают весь открытый текст, начиная с этого блока.

– Возможна подмена последних битов сообщения.

– Отлично подходит для шифрования файлов.

Поскольку *последний блок* шифротекста зависит от всех блоков открытого текста, то его можно использовать для контроля целостности и аутентичности сообщения. Его называют *кодом аутентификации сообщения (MAC – Message Authentication Code).*

Обратная связь по шифротексту (*Cipher-feedback, CFB*) или режим гаммирования с обратной связью (с зацеплением блоков) может использоваться для получения поточного шифра из блочного. Размер блока в данном режиме меньше либо равен размеру блока шифра (рисунок 2.18).

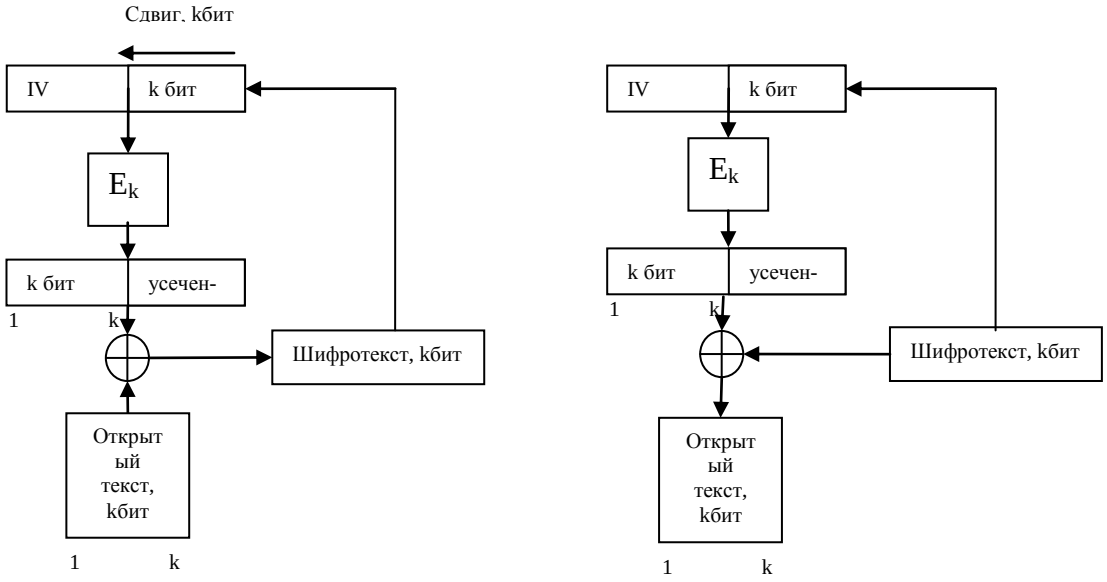


Рисунок 2.18 – Схема обратной связи по шифротексту

Схема работы:

1 Генерируется *IV* и передается перед сеансом связи получателю.

2 Значение *IV* шифруется.

3 Берутся первые *k* крайних слева бит зашифрованного *IV* и складываются (XOR) с *k* битами открытого текста. Получается блок шифротекста из *k* бит.

4 Значение *IV* сдвигается на *k* битов влево, а вместо него становится значение шифротекста.

5 Повторяется пункт 2 до конца цикла шифрования.

Расшифрование происходит аналогично.

Основные характеристики режима:

– Распространение ошибки на весь последующий текст.

– Шифрование каждого блока зависит от всех предыдущих блоков.

– Ошибка в бите шифротекста влияет на текущий и определенное число следующих блоков открытого текста, затем ошибка самоустраняется.

– Шифр самовосстанавливается после потери (добавления) бита шифротекста. Возможна подмена последних битов сообщения.

– Применяется, как правило, для шифрования потоков информации типа оцифрованной речи, видео.

Обратная связь по выходу ***(****Output-feedback, OFB)*, или режим гаммирования позволяет получать поточный шифр в его классическом виде, в отличие от режима CFB в нем отсутствует связь с шифротекстом. Организован аналогично CFB, но IV заполняется не битами шифротекста, а битами, выходящими из-под усечения (рисунок 2.19). В OFB генерируется ключевой поток $K_1 .. K_к$, по к бит за один раз, который складывается с открытым текстом. То есть, для любого блока длины *k* операция зашифрования выглядит следующим образом:

$$C_i = M_i \oplus K_i,$$

где $K_i = E_k(K_{i-1})$, $K_0 = IV$, $i = 1..k$.

Алгоритм дешифрования полностью совпадает с алгоритмом шифрования, что облегчает реализацию режима.

Основные характеристики режима:

– Главное свойство шифра: нет распространения ошибок – неправильный бит шифротекста приведет к одному неправильному биту открытого текста. Единичные ошибки не распространяются.

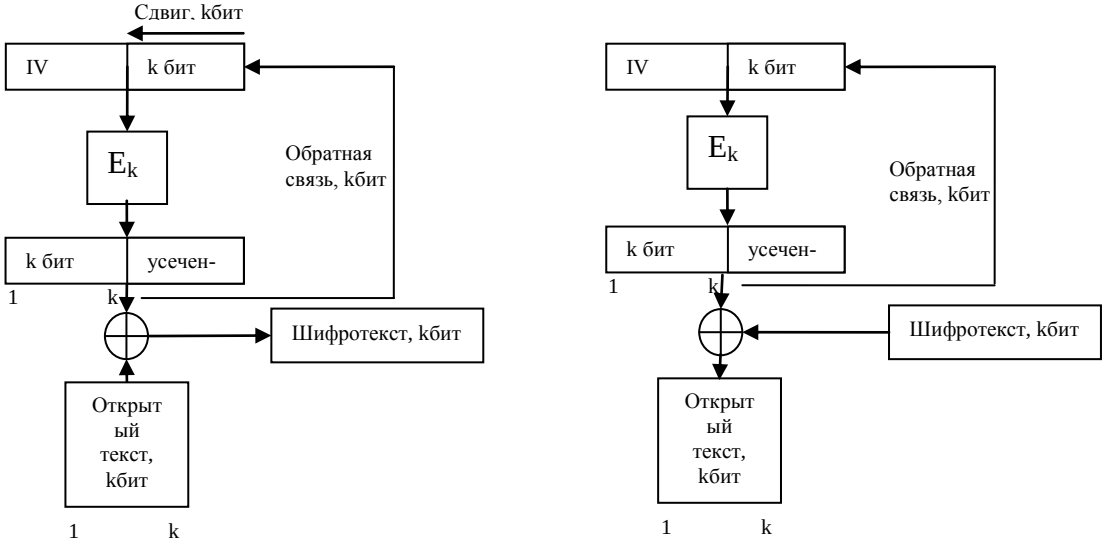


Рисунок 2.19 – Схема обратной связи по выходу

– Единица зашифрованных данных может быть меньше размера блока шифра (что не рекомендуется).

– Потеря добавление бита шифротекста портит весь открытый текст с этого места.

– Область применения: потоки видео, аудио или данных, для которых необходимо обеспечить оперативную доставку. Широко используется у военных наряду с поточными шифрами.

2.3 Асимметричные криптографические алгоритмы

В алгоритмах шифрования с открытым ключом один ключ (называемый открытым ключом) используется для шифрования сообщения, а другой (называемый закрытым ключом) используется для дешифрования сообщения. Шифрование с открытым ключом важно, потому что решены проблемы, связанные с распространением ключей. При симметричном шифровании ключей каждый пользователь для общения друг с другом должен получить копию секретного ключа.

Если этот ключ будет потерян или скопирован, то злоумышленник сможет расшифровать все сообщения. Благодаря шифрованию с открытым ключом пользователь может свободно распространять свой открытый ключ во всем мире, и только он может расшифровать сообщения, зашифрованные этим открытым ключом.

Следует отметить, что знание математики в основе этого или любого другого криптоалгоритма не критично для большинства специалистов по разработке ПО. Но некоторые разработчики, реализующие ПО криптографической защиты КС будут заинтересованы в углублении в криптографию.

2.3.1 Математические основы ассиметричных методов

Между открытым и закрытым ключами существует определенная математическая связь. Необходимо, чтобы по известному открытому ключу, не зная секретной части алгоритма, было бы невозможно (вычислительно трудоемко) получить закрытый ключ шифрования. Для этого в теории криптографии с открытым ключом используются вычислительно трудоемкие задачи факторизации, дискретного логарифмирования, проецирования точек на эллиптической кривой и т.д. Фактическая математика, лежащая в основе создания и применения ключей, будет различаться между различными асимметричными алгоритмами. Многие алгоритмы с открытым ключом в какой-то степени зависят от больших простых чисел, факторизации и теории чисел [15, 17].

Объединяет все эти задачи то, что они используют операцию получения остатка от целочисленного деления.

Прежде чем углубиться в RSA, нужно изучить несколько основных математических понятий из теории чисел.

Простые числа: простое число делится само на себя и на 1. Например, 2, 3, 5, 7, 11, 13, 17 и 23 – простые числа. (Заметим, что само число 1 считается частным случаем и не считается простым).

Взаимно-простые числа: это на самом деле не означает простые числа, это означает, что два числа не имеют общих делителей. Так, например, делители числа 8 (исключая частный случай 1) равны 2 и 4. делителем числа 9 является только 3. Числа 8 и 9 не имеют общих делителей. Они являются взаимно-простыми числами.

Функция Эйлера: определяет число целых чисел, меньших n, которые взаимно просты с n. Гаусс ввёл обозначение $\varphi(n)$, ставшее стандартным. Рассмотрим число 10. Поскольку 2 – делитель 10, он не является взаимно-простым с 10. Но 3 является взаимно-простым с 10. Число 4 не является взаимно-простым, так как оба 4 и 10 имеют делитель 2. Число 5 не является, так как оно является делителем 10. 6 также на взаимно-простое, поскольку оба 6 и 10 имеют 2 в качестве общего делителя. Число 7 является взаимно-простым с 10. Число 8 не взаимно-простые, так как 8 и 10 имеют общий делитель 2. Число 9 является взаимно-простым с 10. Таким образом, числа 3, 7 и 9 взаимно просты с 10. Добавляя 1 как частный случай, функция Эйлера от 10 равна 4.

Леонард Эйлер доказал, что если число n является простым числом, то $\varphi(n) = n - 1$. Таким образом, $\varphi(7) = 6$, $\varphi(13) = 12$.

Леонард Эйлер также доказал, что если число является произведением двух простых чисел (например, p и q), таких как 5 и 7, то функция Эйлера произведения этих двух чисел (в данном случае 35) равна $(p - 1)^* (q – 1)$ (т.е. $\varphi(35) = (5-1) * (7-1) = 24$).

Модуль. С точки зрения программиста, операция модуля состоит в целочисленном делении двух чисел, в качестве результата берется только остаток. Программисты часто используют символ % для обозначения операций по модулю. Таким образом, 10%3 равно 1. Остаток деления 10 на 3 равен 1.

Более формально. Если a является целым, а n - положительным, то $a \bmod n$ определяется как остаток от деления a на n. Таким образом, для любого целого a можно записать

$$a = \text{int}(a / n) * n + (a \bmod n).$$

где $int(x)$ обозначает наибольшее целое число, не превышающее x.

Множество чисел по модулю простого числа p: (0,1, …, p-1) называют конечным полем, обозначают Z_p (или GF(p), Z/pZ). Группой называют множество чисел для которых определена одна операция. Z_p состоит из аддитивной группы и мультипликативной группы. Наиболее современные асимметричные криптосистемы основаны на эллиптических кривых над конечным полем – конечные абелевы группы, в которых можно ставить задачу о вычислении дискретных логарифмов.

2.3.2 Генерация простых чисел

Шифрование с открытым ключом требует работы с большими числами, т.е. с числами, содержащими несколько сотен разрядов точности. Как правило, для большинства языков программирования уже разработаны специальные библиотеки для работы с такими числами. Одной из подзадач открытого шифрования является генерация больших простых чисел длиной в 2000-4000 бит. Базовым методом генерации таких чисел является генерация случайного числа и проверка его на простоту с помощью вероятностных тестов.

Тест простоты Ферма натурального числа n в теории чисел основан на малой теореме Ферма:

Если n – простое число, то оно удовлетворяет сравнению

$$a^{n-1}=1 \pmod{n}, \qquad (2.1)$$

где n не делит a.

Выполнение сравнения (2.1) является необходимым, но не достаточным признаком простоты числа. То есть, если бы для одного $a^{n-1} \neq 1 \pmod{n}$, то число n составное, в противном случаи ничего сказать нельзя, хотя вероятность того, что число является простым увеличивается.

Если для составного числа n выполняется сравнение (2.1), то число n называют *псевдопростым по базе a*. При проверке числа на простоту тестом Ферма выбирают *несколько чисел a.* Чем больше количество a, для которых (2.1), тем больше вероятность, что число n простое. Однако существуют составные числа, для которых сравнение (2.1) выполняется для всех *a взаимно простых с n* – это числа

Кармайкла – 561, 1105 и таких чисел бесконечное множество. Тем не менее, тест Ферма довольно эффективен для обнаружения составных чисел.

Тест Миллера – Рабина – вероятностный полиноминальный тест простоты. Алгоритм был разработан Гари Миллером в 1976 году и модифицирован Майклом Рабином в 1980 году. Тест Миллера – Рабина позволяет *эффективно* определять, является ли данное число составным, обходит проблему с числами Кармайкла. Тест принимает составное число за простое с вероятностью ¼ для любого случайного основания a , поэтому циклическое выполнение теста позволяет уменьшить вероятность ошибки до любой малой величины. Однако, с его помощью также, как и с помощью теста Ферма, нельзя строго доказать простоту криптографии для получения больших случайных простых чисел.

Пусть m – нечетное число большее 1. Число m - 1 однозначно представляется в виде $m - 1 = 2^s * t$, где t нечетно. Целое число a, $1<a<m$, называется свидетелем простоты числа m, если выполняется одно из условий:

$$a=1 \pmod m$$

или существует целое число k, $0 \le k < s$, такое, что $a^{2ktt} = m-1 \pmod m$.

Теорема Рабина утверждает, что составное нечетное число m имеет не более $\varphi(m)/4$ различных *свидетелей простоты*, где $\varphi(m)$ – функция Эйлера.

Алгоритм Миллера – Рабина параметризуется количеством раундов r. Рекомендуется брать r порядка величины $log_2(m)$, где m – проверяемое число.

Для данного m находятся такие целое число s и целое нечетное число t, что $m-1=2^s * t$. Выбирается случайное число a, $1<a<m$. Если a не является свидетелем простоты числа m, то выдается ответ *«m составное»*, и алгоритм завершается. Иначе, выбирается новое случайное число a и процедура проверки повторяется. После нахождения r свидетелей простоты, выдается ответ *«m, вероятно, простое»*, и алгоритм завершается.

Алгоритм может быть записан на псевдокоде следующим образом:

Ввод: $m>2$, нечетное натуральное число, которое необходимо проверить на простоту;

r – количество раундов.

Вывод: *составное*, означает, что *m* является составным числом;

вероятно простое, означает, что *m* с высокой вероятностью является простым числом.

Представить *m*-1 в виде $2^s * t$, где *t* нечетно, можно сделать последовательным делением *m*-1 на 2.

цикл A: повторить *r* раз:

Выбрать случайное целое число *a* в отрезке [2, *m*-2]

$$x \leftarrow a^t \bmod m$$

если x=1 или x=m-1, то перейти на следующую итерацию цикла A

цикл B: повторить *s*-1 раз

$$x \leftarrow x^2 \bmod m$$

если *x*=1, то вернуть составное

если *x*=*m*-1, то перейти на следующую итерацию цикла A

вернуть составное

вернуть вероятно простое

Из теоремы Рабина следует, что если *r* случайно выбранных чисел оказались свидетелями простоты числа *m*, то вероятность того, что *m* составное, не превосходит 4^{-r}.

Пример кода на языке C# с большими числами для реализации теста Ферма приведен на рисунке 2.20.

```
for(int round = 0; round < confidence; round++)
{
    // генерация большого случайного числа: a < n
    a.genRandomBits(testBits, rand);
    // вычисление a^(p) mod p, если thisVal = p
    BigInteger expResult = a.modPow(thisVal, thisVal);
    //если a^(p) mod p != a, то составное
    if(expResult != a)
        return false;
}
return true;
```

Рисунок 2.20 – Тест Ферма

Генерация псевдопростых чисел на языке C# может быть выполнена с помощью встроенной библиотеки (рисунок 2.21).

Для измерения времени выполнения алгоритма на языке C# можно воспользоваться классом Environment (рисунок 2.22).

```
Console.Write(
        "\nGenerating 512-bits random pseudoprime. . .");
    Random rand = new Random();
    BigInteger prime = BigInteger.genPseudoPrime(512, 5, rand);
    Console.WriteLine("\n" + prime);
```

Рисунок 2.21 – Генерация псевдопростых чисел

```
int dwStart = System.Environment.TickCount;
Console.WriteLine(System.Environment.TickCount - dwStart);
```

Рисунок 2.22 – Замер времени

2.3.3 Криптосистема RSA

Одной из первых криптосистем с открытым ключом, еще актуальных сегодня является криптосистема RSA, разработанная в 1977 году и названная в честь ее создателей: Рона Ривеста, Ади Шамира и Леонарда Эйдельмана (*R*ivest, *S*hamir и *A*ldeman).

В этой криптосистеме блоки сообщений открытого текста и шифрованного текста обрабатываются как большие числа, представленные тысячами бит. Шифрование и дешифрование выполняются с использованием модульной экспоненты, а правильность этих алгоритмов шифрования и дешифрования основана на теореме Эйлера и других свойствах модульной арифметики. Криптостойкость обусловлена сложностью разложения больших целых чисел на простые сомножители – задачей *факторизации*.

Возможность гарантированно оценить защищенность алгоритма RSA позволила использовать алгоритм RSA в банковских компьютерных сетях, в частности для работы с удаленными клиентами (обслуживание кредитных карточек).

В настоящее время алгоритм RSA используется во многих стандартах, среди которых SSL, S-HHTP, S-MIME, SWAN, STT и PCT.

Алгоритм создания открытого и секретного ключей выполняет получатель:

1) сгенерировать два больших случайных простых числа p и q примерно равного размера, таких чтобы размер их произведения был бы 2048 бит, 4096 бит и т. д.;

2) вычислить произведение $n=p*q$ и функцию Эйлера $\varphi(n)=(p-1)(q-1)$;

3) сгенерировать целое число e: $1<e<\varphi(n)$, т.е. взаимно простое с $\varphi(n)$;

Пара чисел (e,n) публикуется в качестве *открытого ключа*.

4) вычислить целое число d, удовлетворяющее условию:

$$e*d=1(\text{mod } \varphi(n)), \qquad (2.2)$$

используя расширенный алгоритм Евклида.

Пара чисел (d,n) является *секретным ключом*.

Шифрование сообщения с использованием открытого ключа выполняет отправитель:

1 M – сообщение представить блоками и закодировать целыми числами в интервале от 0 до n-1: $\{m_i, i=0,1,\ldots, |m_i|<=n-1\}$, т.е. размером $k=[\log_2(n)]$ бит каждое;

2 *зашифровать* сообщение M:

$$c_i=m_i^e \bmod(n), i=0,1,\ldots, \qquad (2.3)$$

Дешифрование полученного криптотекста C с использованием секретного ключа выполняет получатель*:*

расшифровать, полученное сообщение C:

$$m_i=c_i^d \bmod (n), i=0,1,\ldots, \qquad (2.4)$$

Безопасность криптосистемы RSA

Безопасность криптосистемы RSA основана на трудности нахождения секретного d, зная открытые e и n. Если станет известным значение функции Эйлера $\varphi(n) = (p - 1)(q - 1)$, то будет легко вычислить секретный ключ d из e.

Таким образом, пользователю нужно держать p и q в секрете (или даже уничтожать все знания о них), так как любой, кто узнает значения p и q, сразу узнает значение $\varphi(n)$. Любой, кто узнает значение $\varphi(n)$, может вычислить секретный ключ d, $d = e^{-1} \bmod \varphi(n)$, используя расширенный алгоритм Евклида.

Таким образом, безопасность криптосистемы RSA тесно связана с трудноразрешимой *задачей факторизации* числа n, которая дает значения p и q. Поскольку эта проблема трудноразрешима, то можно полагаться на безопасность криптосистемы RSA, при условии использования достаточно большого модуля. По состоянию на 2019 год рекомендуется использовать 2048-битный модуль.

Также были продемонстрированы атаки с использованием побочных каналов (дополнительных каналов информации о системе) на RSA на основе измерения времени, затраченного на дешифрование и / или энергопотребление CPU, выполняющего операцию. Следует также позаботиться о правильном использовании криптосистемы RSA из-за ее детерминированного характера. Например, предположим, что используется алгоритм RSA для шифрования двух сообщений открытого текста, M1 и M2, в соответствующие зашифрованные тексты C1 и C2 с использованием одного и того же открытого ключа. Поскольку RSA является детерминированным, то в этом случае, если C1 = C2, то M1 = M2. К сожалению, этот факт может позволить криптоаналитику выводить информацию из криптотекстов, полученных из якобы разных открытых текстов. Криптосистема Эль-Гамаля не имеет такого же недостатка.

Эффективная реализация криптосистемы RSA

Реализация криптосистемы RSA требует эффективных алгоритмов для следующих задач [40, 44]:

– Проверка чисел на простоту, то есть проверка является ли целое число простым. Этот алгоритм используется на этапе настройки для выбора чисел p и q модуля RSA. Каждый простое число выбирается путем создания серии случайных чисел и остановки, как только будет найдено простое.

– Вычисление наибольшего общего делителя, который используется на этапе настройки для выбора показателя e шифрования.

– Вычисление обратного модуля, которое используется на этапе установки для вычисления показателя дешифрования d с учетом показателя шифрования e.

– Модульная экспонента, используемая в алгоритмах шифрования и дешифрования.

Очевидно, что сначала вычисление экспоненты, а затем применение оператора *mod* неэффективно, так как мощность может быть очень большим числом.

Для программной реализации ассиметричных алгоритмов на языке C# можно использовать класс RSACryptoServiceProvider.

Пример шифрования приведен на рисунке 2.23.

```
byte[] RSAEncrypt(byte[] DataToEncrypt, RSAParameters RSAKeyInfo, bool DoOAEPPadding){
        using (var RSA = new RSACryptoServiceProvider()) {
                byte[] encryptedData;
                RSA.ImportParameters(RSAKeyInfo) // Импорт ключей
        // Шифрование данных
                encryptedData = RSA.Encrypt(DataToEncrypt, DoOAEPPadding);
                return encryptedData;
        }}
```

Рисунок 2.23 – Шифрование встроенными средствами C#

Расшифрование (рисунок 2.24):

```
byte[] RSADecrypt(byte[] DataToDecrypt, RSAParameters RSAKeyInfo, bool DoOAEPPadding){
        using (var RSA = new RSACryptoServiceProvider())     {
                byte[] decryptedData;
                RSA.ImportParameters(RSAKeyInfo) // Импорт ключей
                        // Расшифрование данных
                decryptedData = RSA.Decrypt(DataToEncrypt, DoOAEPPadding);
                return decryptedData;
        }}
```

Рисунок 2.24 – Расшифрование встроенными средствами C#

Информация о ключах может быть передана из провайдера в провайдер при помощи структуры RSAParameters. Объект RSAParameters может быть получен из криптопровайдера при помощи функции RSACryptoServiceProvider.ExportParameters().

2.3.4 Криптосистема Эль-Гамаля

Алгоритм шифрования Эль-Гамаля (Elgamal) так же как RSA, основан на применении больших чисел для генерации открытого и закрытого ключа, криптостойкость же обусловлена сложностью вычисления дискретных логарифмов.

Криптосистема Эль-Гамаля, названная в честь ее изобретателя, Тахера Эль-Гамаля, является криптосистемой с открытым ключом, которая использует

рандомизацию, так что независимые шифрования одного и того же открытого текста могут создавать разные зашифрованные тексты.

Алгоритм шифрования Эль-Гамаля основан на представлении входных блоков текста в виде чисел и применении арифметических операций на этих числах для выполнения шифрования и дешифрования. Перед описанием подробностей для этой криптосистемы в конечных полях, будут рассмотрены некоторые связанные понятия из теории чисел.

В системе чисел *Zp* вся арифметика выполняется по модулю простого числа, *p*. Число, *g* в *Zp*, называется *генератором* или *примитивным корнем по модулю p*, если для каждого положительного целого *i* из *Zp* существует целое число *k* такое, что $i = g^k \bmod p$.

Существуют функции Эйлера $\varphi(\varphi(p)) = \varphi(p - 1)$ для *Zp*. Поэтому можно тестировать разные числа, пока не найдется то, которое является генератором. Чтобы проверить, является ли число *g* генератором, достаточно проверить, что $g^{(p-1)/pi} \bmod p \; != 1$ для каждого простого множителя *pi* для $\varphi(p) = p - 1$.

Если число не является генератором, одна из этих степеней будет равна 1. Как правило, было бы сложно определить такое *p*-1, чтобы можно быстро найти все его простые сомножители. Но можно сделать эту работу легко, выбирая кандидатов для простого числа *p* исходя из знания факторизации *p*-1.

Для криптосистемы Эль-Гамаля требуется генератор, поэтому допустим здесь, что можно выбрать любое простое число *p*, и быстро находить генератор *g* для *Zp*. Как только станет известен генератор *g*, можно эффективно вычислить $x = g^k \bmod p$ для любого значения *k*.

Обратно, если заданы *x*, *g* и *p*, то задача определения *k* такого, что $x = g^k \bmod p$ называется *проблемой дискретного логарифма*. Как и факторизация, проблема дискретного логарифма относится к сложным.

Безопасность криптосистемы Эль-Гамаля зависит от сложности проблемы дискретного логарифма [15, 17, 38, 44].

Схема применения криптосистемы Эль-Гамаля

1 Этап подготовки.

1) ПользовательБ выбирает случайное большое простое число p и находит генератор g для Zp.

2) Затем выбирает случайное число x между 1 и p - 2, и вычисляет

$$y = g^x \bmod p.$$

3) Число, x, является секретным ключом ПользователяБ. Тройка (p, g, y) является его открытым ключом.

2 Этап шифрования сообщения.

Когда ПользовательА хочет зашифровать сообщение с открытым текстом M для ПользователяБ, то он:

1) Получает открытый ключ ПользователяБ (p, g, y).
2) Генерирует случайное число k в интервале 1 и $p - 2$.
3) Используя модульное умножение и экспоненту, вычисляет два числа:

$$a = g^k \bmod p,$$

$$b = My^k \bmod p.$$

4) Полученная пара чисел (a,b) является криптограммой сообщения M.

3 Этап расшифрования сообщения.

1) ПользовательБ получает по открытым каналам криптограмму (a, b).
2) Вычисляет M = (b / a^X) mod p.

Существует аналогичная система эллиптических кривых.

2.3.5 Хэш-функции

Криптографическая хеш-функция служит для создания сжатого дайджест, а также обеспечивает отображение, которое является детерминированным, односторонним и устойчивым к коллизиям.

Свойства и применение [27, 39, 44].

Одним из важнейших свойств криптографических хеш-функций является то, что они являются *односторонними*. То есть, из сообщения M должно быть легко вычислить значение хеша, $H(M)$. Однако, из известного значения хеша x, будет вычислительно трудно найти сообщение M, такое, что $x = H(M)$.

Кроме того, значение хеш-функции должно быть значительно меньше, чем типичное сообщение. Например, обычно используемая стандартная хеш-функция SHA-256 создает хеш-значения с 256 битами. Эта хеш-функция использует несколько методов, применяемых в симметричном шифровании, включая подстановку, перестановку, исключающее-или итерацию таким образом, чтобы обеспечить настолько большую диффузию ввода, что изменение любого бита на входе может потенциально повлиять на значение каждого бита на выходе.

Сопротивление коллизиям.

Хеш-функция H – это отображение входных строк в меньшие выходные строки.

Говорят, что H имеет *слабое сопротивление коллизиям*, если для любого сообщения M трудно вычислить другое сообщение M' != M, такое, что

$$H(M') = H(M).$$

Хэш-функция H имеет *сильное сопротивление коллизиям*, если вычислительно трудно найти два разных сообщения, M_1 и M_2, так что $H(M_1) = H(M_2)$. То есть при слабом сопротивлении столкновению пытаются избежать коллизии с определенным сообщением, а при сильном сопротивлении коллизиям стараются избегать коллизий в целом. Обычно сложно доказать, что реальные криптографические хеш-функции имеют сильное сопротивление коллизиям, поэтому криптографы обычно предоставляют экспериментальные доказательства этого свойства.

Конструкция Меркля-Дамгарда (Merkle-Damgard)

Обычной структурой для хеш-функции является использование в качестве строительного блока *криптографической функции сжатия* $C(X, Y)$, которая является криптографической хеш-функцией C, принимающей в качестве входных данных две строки, X и Y, где X имеет фиксированную длину m, а Y имеет фиксированную длину n, и создает хеш-значение длины n.

Сообщение M разделяют на несколько блоков M_1, M_2,..., M_k, каждый длиной m, где последний блок дополняется однозначным образом дополнительными битами, чтобы сделать его длиной m.

Начинают с применения функции сжатия C к первому блоку, M_1, и фиксированной строки v длиной n, известной как *вектор инициализации*. Обозначив результирующее хеш-значение с помощью $d_1 = C(M_1, v)$. Далее применяют функцию сжатия к блоку M_2 и d_1, в результате чего получают хеш-значение $d_2 = C(M_2, d_1)$ и так далее.

Определяют значение хеша сообщения H как равное d_k.

Этот метод построения криптографической хеш-функции из криптографической функции сжатия, показанной на рисунке 2.25, известен как конструкция Меркля-Дамгарда, после его изобретателей Ральфа Меркля и Ивана Дамгарда.

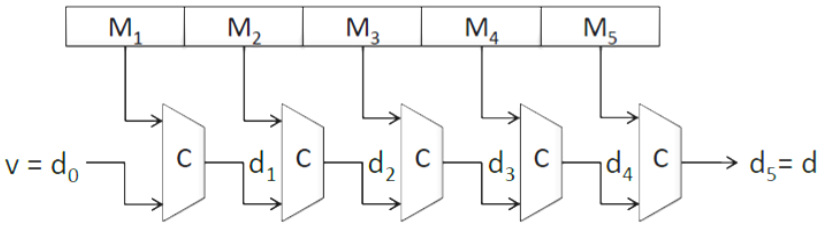


Рисунок 2.25 - Конструкция Меркля-Дамгарда

В конструкции Меркля-Дамгарда, если злоумышленник обнаружит коллизии между двумя различными сообщениями, M_1 и M_2, т. е. $H(M_1) = H(M_2)$, то он может сформировать другую произвольную коллизию, действительно, для любого сообщения P выполняется $H(M_1||P) = H(M_2||P)$, где символ «||» обозначает конкатенацию строк. Таким образом, важно, чтобы функция сжатия имела сильное сопротивление коллизиям.

Практические хэш-функции для криптографических приложений.

Хеш-функциями, рекомендуемыми в настоящее время для криптографических приложений, являются функции SHA-256 и SHA-512, стандартизированные NIST, где SHA обозначает «алгоритм защищенного хеширования», а числовой суффикс относится к длине значения хеша. Эти функции выполняются в соответствии со структурой Меркля-Дамгарда

SHA-256 использует функцию сжатия с входами m = 512 бит и n = 256 бит и выдает значения хеш-функции n = 256 бит. Для SHA-512 этими параметрами являются m = 1024 и n = 512.

Пример программной реализации SHA-256 на языке C# с использованием встроенной библиотеки System.Security.Cryptography (рисунок 2.26).

```
static string SHA256(string m) {
            var crypt = new System.Security.Cryptography.SHA256Managed();
            var hash = new System.Text.StringBuilder();
            byte[] crypto = crypt.ComputeHash(Encoding.UTF8.GetBytes(m));
            foreach (byte theByte in crypto)  {
                hash.Append(theByte.ToString("x2"));
            }
            return hash.ToString();
}
```

Рисунок 2.26 – SHA-256 в библиотеке Cryptography

Хеш-функция MD5, где MD относится к «дайджесту сообщений», по-прежнему используется в устаревших приложениях. Тем не менее, он считается небезопасным, поскольку были продемонстрированы несколько атак против него.

2.3.6 Цифровые подписи

Цифровая подпись (ЦП) – это способ доказательства подлинности сообщения на основе привязки идентичности пользователя к этому сообщению.

Общая схема (модель) ЦП заключается в том, что одна сторона, отправитель, использует алгоритм подписи со своим закрытым ключом для *формирования* цифровой подписи $S(M)$ на сообщении M, представляющей собой строку бит [9]. Другая сторона, получатель, осуществляет *проверку подписи* отправителя на M, используя только открытый ключ отправителя, сообщение M и подпись отправителя $S(M)$ (см. рисунок 2.27). Также схема ЦП включает генерацию ключей для постановки и проверки подписи [9, 39, 48].

Два важных свойства для схемы с цифровой подписью:

– Не подделываемость. Злоумышленнику должно быть трудно подделать подпись $S(M)$, для сообщения M.

– Неизменяемость. Для злоумышленника должно быть трудно подпись $S(M)$ для сообщения M преобразовать в действительную подпись для другого сообщения, N.

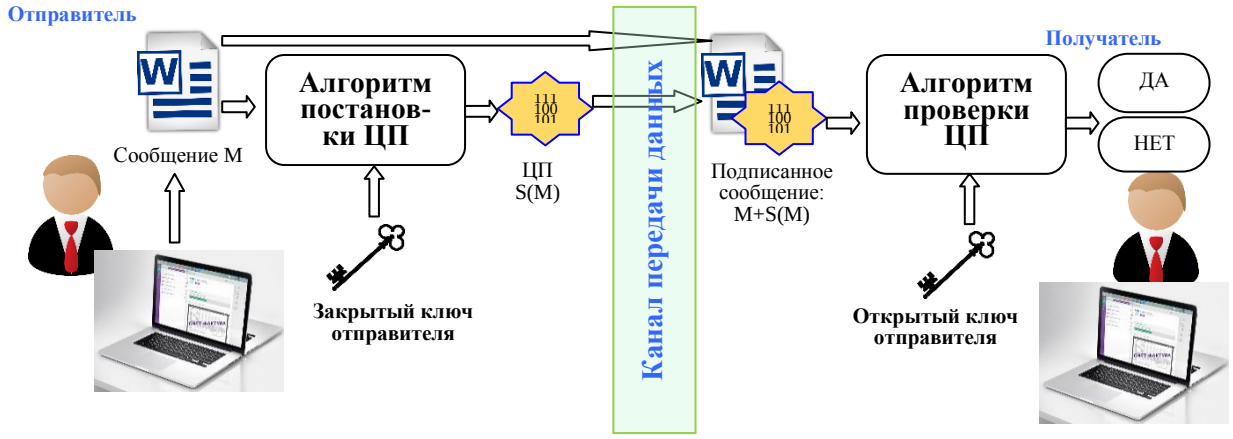


Рисунок 2.27 – Процесс постановки и проверки цифровой подписи

Если схема цифровой подписи достигает этих свойств, то на самом деле она достигает еще одного неотказа. Пользователю должно быть трудно заявить, что он не подписывал документ, M, после постановке цифровой подписи, $S(M)$, на этот документ.

Выделяют подходы к созданию схем ЦП:

– с помощью симметричных криптосистем – специальные режимы функционирования (первые варианты);

– на основе ассиметричной криптосистемы;

– схемы со специально разработанными алгоритмами вычисления и проверки подписи.

Схемы современных ЦП, являющихся ЦП на основе ассиметричной криптосистемы, подразделяют на два вида [38]:

1 Схема ЦП с восстановлением текста. Пара преобразований: шифрование с закрытым ключом и расшифрование с открытым ключом обозначено как (E,D). Постановка ЦП S осуществляется через D:

$$S=D(M).$$

Проверка осуществляется через E:

$$E(S)=M.$$

Должны выполнятся требования:

- $M=E(D(M))$ для всех M.
- $D(M)$ для заданного M.

2 Схема ЦП с хэш-функцией (ЦП с дополнением). ЦП ставится не на само сообщение M, а на его хеш-значение:

$$h(M)\ ;\ S=D(h(M))$$

Классическими примерами ЦП на основе ассиметричной криптосистемы являются подписи RSA и Эль-Гамаля. Криптостойкость современных схем ЦП обусловлена сложностью решения задачи дискретного логарифмирования в группе точек эллиптической кривой, а также криптостойкостью хэш–функции, применяемой в данной ЦП [9].

2.3.6.1 Схема подписи RSA и Elgamal

В схеме подписи RSA, алгоритм RSA применяется по тем же формулами (2.2)-(2.4), но в обратном направлении. Отправитель для постановки ЦП шифрует сообщение M, используя свой *секретный ключ* d, а получатель проверяет ЦП открытым ключом отправителя.

В схеме Elgamal подписи документов выполняются путем рандомизации, как в шифровании Elgamal, но детали подписей Elgamal отличаются от шифрования Elgamal.

Подписью к сообщению M является решение (a, b) уравнения

$$g^M \equiv y^r a^b \ (\mathrm{mod}\ p),$$

где p – простое число,

g – первообразный корень $y = g^x \bmod p,\ x \in Z_{p-1}$.

Открытым ключом является тройка (p, g, y), секретным ключом – x.

Чтобы подписать сообщение M, отправитель генерирует новое одноразовое случайное число k, $k \in Z^*_{p-1}$ и вычисляет следующие два числа:

$$a = g^k \bmod p;$$

$$b = k^{-1} (M - xa) \bmod (p-1).$$

Пара (a, b) является подписью отправителя на сообщении M.

Чтобы проверить подпись (a, b) на M, получатель проверяет выполняется ли равенство:

$$y^a a^b \bmod p = g^M \bmod p \ ?$$

Безопасность этой схемы основана на том факте, что вычисление b зависит как от случайного числа k, так и от секретного ключа отправителя x. Кроме того, поскольку k является случайным, его обратное также является случайным; следовательно, злоумышленник не может отличить b от случайного числа, если только он не может решить проблему дискретного логарифма, чтобы определить число k из a (что равно $g^k \bmod p$). Таким образом, подобно шифрованию Эль-Гамаля, безопасность схемы подписи Эль-Гамаля основана на сложности вычисления дискретных логарифмов.

Кроме того, важно, чтобы отправитель никогда не использовал случайное число k для двух разных подписей.

2.3.6.2 Использование хеш-функций в цифровых подписях

Применение алгоритмов RSA и Elgamal, приведенных выше, для цифровой подписи непосредственно к исходным сообщениям не подходит для практического применения. Во-первых, обе схемы неэффективны, если подписываемое сообщение M очень длинное. Например, создание подписи RSA включает в себя шифрование сообщения M с использованием закрытого ключа, а проверка подписи Elgamal требует модульного возведения в степень M. Во-вторых можно создать действительные подписи RSA на комбинированных сообщениях из существующих подписей RSA.

По практическим соображениям и соображениям безопасности реальные схемы цифровой подписи обычно применяются к криптографическим хэшам сообщений, а не к реальным сообщениям. Пример схемы ЦП с хеш-функцией на основе RSA приведен на рисунке 2.28.

Такой подход значительно снижает риск изменчивости для сигнатур RSA, например, поскольку крайне маловероятно, что произведение двух значений хеша,

$H(M)$ и $H(N)$, будет равно хешу произведения сообщений, $M * N$. Более того, подпись значения хеш-функции более эффективна, чем подпись полного сообщения.

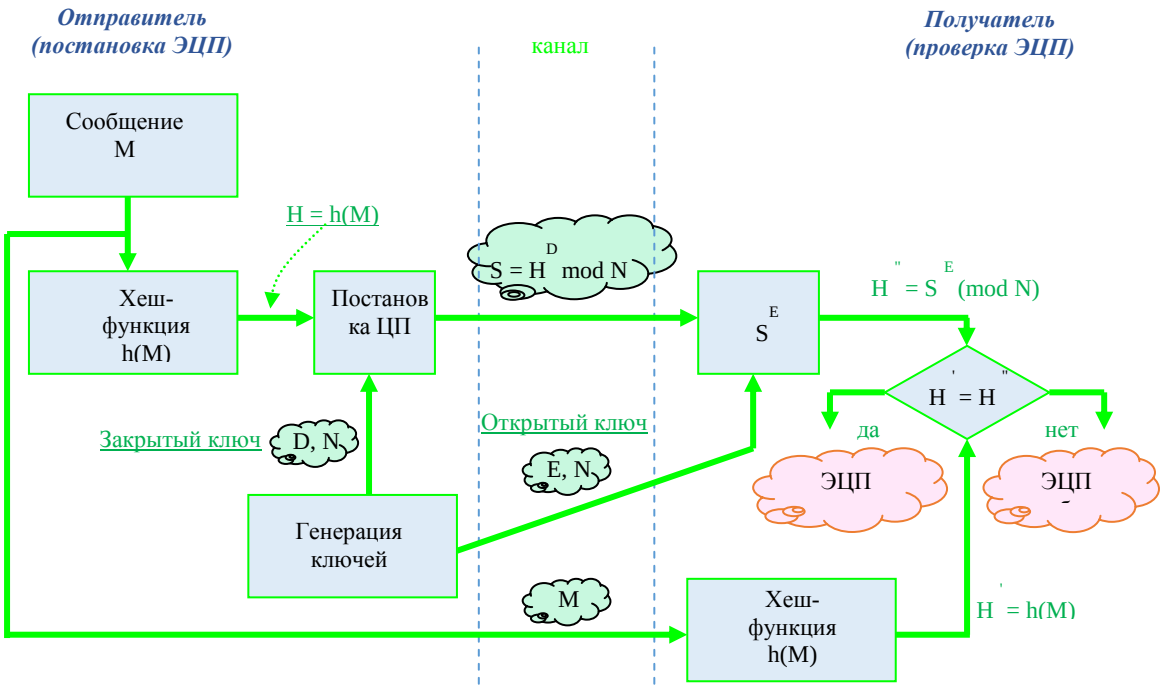


Рисунок 2.28 – Схема ЦП RSA с хеш-функцией

В такой схеме безопасность цировой подписи хеш-значений зависит как от безопасности используемой схемы подписи, так и от безопасности используемой криптографической хеш-функции. Например, предположим, что злоумышленник обнаружил коллизию между двумя сообщениями, M и N, относительно хэш-функции H, так что $H(M) = H(N)$.

Если тогда злоумышленник сможет заставить пользователя подписать хеш $H(M)$ сообщения M, то он фактически обманет пользователя, подписавшего сообщение M.

2.4 SSL / TLS

Какое шифрование используется на банковских сайтах и электронной коммерции? В общем, симметричные алгоритмы быстрее и требуют более короткой

длины ключа, чтобы быть столь же безопасными, как асимметричные алгоритмы. Тем не менее, существует проблема безопасного обмена ключами. Большинство решений для электронной коммерции используют асимметричный алгоритм для обмена симметричными ключами, а затем используют симметричные ключи для шифрования фактических данных. При посещении веб-сайтов с HTTPS, а не в HTTP (S обозначает безопасный), трафик между вашим браузером и веб-сервером будет зашифрован. Обычно это делается с использованием SSL (Secure Sockets Layer) или TLS (Transport Layer Security). SSL и TLS являются асимметричными системами. SSL, более старая из двух технологий, используемая для обеспечения безопасности транспортного уровня с помощью шифрования с открытым ключом. SSL была разработана Netscape для передачи частных документов через Интернет. По соглашению, URL-адреса, для которых требуется SSL-соединение, начинаются с https вместо http [2, 48]. Всего было разработано несколько версий.

Основной процесс установления соединения SSL / TLS показан на рисунке 2.29.

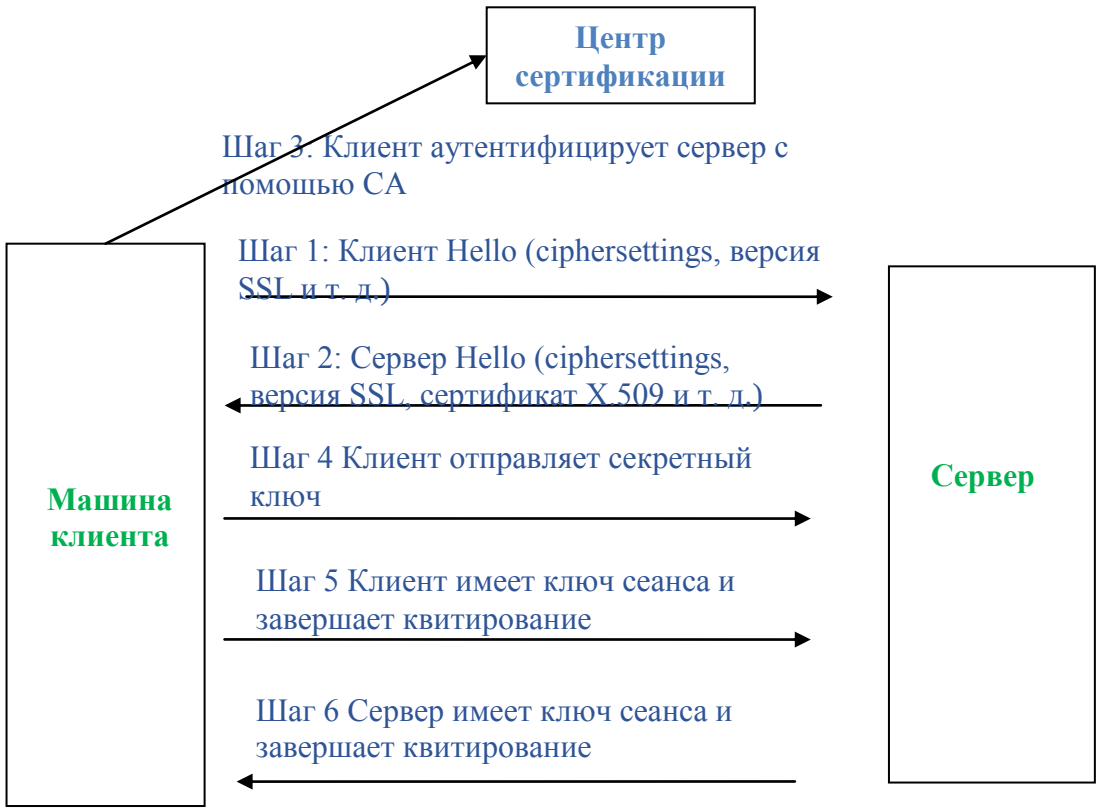


Рисунок 2.29 – SSL / TLS

Процесс включает в себя несколько сложных шагов:

1 Клиент отправляет информацию о своих криптографических возможностях, включающую в себя алгоритмы, на которые он способен, какие алгоритмы хеширования он может использовать для целостности сообщений и связанную с ними информацию.

2 Сервер отвечает, выбирая лучшее шифрование и хэширование, которое клиент и сервер могут выполнить вместе и передает эту информации клиенту. Сервер также отправляет свой собственный сертификат, и если клиент запрашивает ресурс сервера, требующий аутентификации клиента, сервер запрашивает сертификат клиента.

3 Клиент использует информацию, отправленную сервером для аутентификации сервера, проверяя цифровой сертификат в соответствующем центре сертификации. При неудачной проверке, браузер предупреждает пользователя о том, что сертификат не может быть проверен. Если сервер успешно аутентифицирован, клиент переходит к следующему шагу.

4 Используя все данные, сгенерированные в рукопожатии на предыдущих шагах, клиент создает предварительный главный ключ для сессии, шифрует его с помощью открытого ключа сервера, который он получил от сертификата X.509 сервера, а затем отправляет к серверу.

5 Если сервер запросил аутентификацию клиента, сервер также будет проверять сертификат клиента X.509. Это не выполняется на большинстве сайтов электронной коммерции и банковских услуг

6 Клиент и сервер используют главный ключ для генерации сеансовых ключей. Это симметричные ключи (такие как AES), которые будут использоваться на протяжении всего сеанса для шифрования информации между клиентом и сервером.

7 Клиент отправляет на сервер сообщение о том, что будущие сообщения от клиента будут зашифрованы с помощью сеансового ключа.

8 Сервер отправляет клиенту сообщение, информирующее его о том, что будущие сообщения с сервера будут зашифрованы с помощью сеансового ключа.

Этот процесс обеспечивает процесс не только безопасного обмена симметричных ключей, но и проверки сервера и (опционально) проверки клиента. Так обеспечивается безопасный веб-трафик.

Контрольные вопросы

1 Назвать отличия алгоритмов симметричного и асимметричного шифрования.

2 Пояснить основные принципы построения блочных и поточных алгоритмов.

3 Пояснить суть принципов рассеивания и перемешивания.

4 Основные характеристики и алгоритм шифра Магма.

5 Основные характеристики и алгоритм шифра Кузнечик.

6 Для чего разработаны различные режимы работы блочных шифров?

7 Вектор инициализации: понятие и назначение, формирование для различных режимов.

8 В чем заключается алгоритм гаммирования и как реализуется в шифре Магма?

9 Простые числа: понятие, способы проверки и генерации.

10 Алгоритм Миллера – Рабина. Что позволяет определить тест Миллера – Рабина, его параметры.

11 На каких задачах основана стойкость ассимметричных алгоритмов шифрования?

12 Преимущества алгоритма шифрования Эль Гамаля в сравнении с RSA.

13 Какие задачи позволяет решить цифровая подпись?

14 Какую роль играет хеш-функция в реализации цифровой подписи?

15 Шаги процесс установления соединения SSL / TLS.

3 Аутентификация

3.1 Основные понятия аутентификации

Идентификация, аутентификация и авторизация. Большинство сталкивается с этими тремя концепциями каждый день, но не все знают разницу.

Аутентификация – это процесс определения того, действительно ли кто-то или что-то является тем, за кого он себя объявляет. Технология аутентификации обеспечивает контроль доступа для систем, проверяя, соответствуют ли учетные данные пользователя учетным данным в базе данных авторизованных пользователей или на сервере аутентификации данных [7, 38].

Обычно процесс *идентификации* пользователей использует идентификатор пользователя, а аутентификация выполняется, когда пользователь предоставляет учетные данные, например пароль, соответствующий этому идентификатору пользователя. Большинство пользователей наиболее знакомы с использованием пароля, который, как часть информации, которая должна быть известна только пользователю, называется *фактором аутентификации знаний*. Другие факторы аутентификации и способы их использования для двухфакторной или многофакторной аутентификации (MFA) описаны ниже.

Аутентификация в кибербезопасности.

Аутентификация важна, поскольку она позволяет организациям защищать свои сети, разрешая доступ только для аутентифицированных пользователей только к проверенным им ресурсам (или процессам), которые могут включать КС, сети, базы данных, веб-сайты и другие сетевые приложения или службы.

После аутентификации пользователь или процесс обычно подвергаются процессу *авторизации*, чтобы определить, должен ли аутентифицированный субъект иметь доступ к защищенному ресурсу или системе. Пользователь может быть аутентифицирован, но может не получить доступ к ресурсу, если этому пользователю не было предоставлено разрешение на его доступ.

Термины аутентификации и авторизации часто используются взаимозаменяемо; хотя они часто могут быть реализованы вместе, две функции

различны. А*утентификация* – это процесс проверки подлинности зарегистрированного пользователя, происходящий перед разрешением доступа к защищенному ресурсу, *авторизация* – это процесс проверки того, что аутентифицированный пользователь получил разрешение на доступ к запрошенным ресурсам. Процесс, посредством которого доступ к этим ресурсам разрешен или ограничен определенному числу пользователей, называется *контролем доступа*. Процесс аутентификации всегда приходит перед процессом авторизации.

Как используется аутентификация

Аутентификация пользователя происходит в большинстве взаимодействий человека с КС за пределами гостевых учетных записей, автоматически регистрируемых учетных записей. Как правило, пользователь должен выбрать имя пользователя или идентификатор пользователя и предоставить действительный пароль для начала использования системы. Аутентификация пользователя разрешает взаимодействие между машинами и операционными системами и приложениями, а также проводными и беспроводными сетями для обеспечения доступа к сетевым и интернет-подключенным системам, приложениям и ресурсам.

Многие компании используют аутентификацию для проверки пользователей, которые регистрируются на их сайтах. Без правильных мер безопасности пользовательские данные, такие как номера кредитных и дебетовых карт, а также номера социального страхования, могут попасть в руки киберпреступников.

Организации также используют аутентификацию для контроля доступа пользователей к корпоративным сетям и ресурсам, к компьютерам и серверам, к своим приложениям и сетям.

Для предприятий и других крупных организаций аутентификация может быть выполнена с использованием системы единого входа (SSO), которая предоставляет доступ к нескольким системам с одним набором учетных данных для входа.

Как работает аутентификация

Во время проверки подлинности учетные данные, предоставленные пользователем, сравниваются с данными в файле в базе данных пользователя

авторизованных пользователей либо в локальной операционной системе, либо через сервер аутентификации. Если учетные данные совпадают, и аутентифицированный субъект имеет право использовать ресурс, процесс завершается, и пользователю предоставляется доступ. Предоставленные разрешения и папки определяют как среду, которую видит пользователь, так и способ взаимодействия с ней, включая часы доступа и другие права, такие как объем пространства ресурсов.

Традиционно аутентификация осуществлялась системами или ресурсами, к которым осуществляется доступ; например, сервер будет аутентифицировать пользователей, используя свою собственную систему паролей, реализованную локально, используя идентификаторы входа (имена пользователей) и пароли. Предполагается, что знание учетных данных для входа гарантирует подлинность пользователя. Каждый пользователь вначале регистрируется (или зарегистрирован кем-то другим, например системным администратором), используя назначенный или самостоятельно объявленный пароль. При каждом последующем использовании пользователь должен знать и использовать ранее объявленный пароль.

Тем не менее, протоколы приложений в Интернете, HTTP и HTTPS, не имеют состояния, что означает, что при строгой проверке подлинности конечные пользователи проверяются каждый раз, когда они обращаются к ресурсу с помощью HTTPS. Вместо того чтобы обременять конечных пользователей этим процессом для каждого взаимодействия через Интернет, защищенные системы часто полагаются на *аутентификацию на токенах*, в которой аутентификация выполняется один раз в начале сеанса. Система аутентификации выдает подписанный токен аутентификации в приложение конечного пользователя, и этот токен добавляется к каждому запросу от клиента.

Аутентификация подлинности для систем и процессов может выполняться с использованием учетных данных машины, которые работают как идентификатор пользователя и пароль, за исключением того, что учетные данные автоматически отправляются данным устройством. Они могут также использовать цифровые сертификаты, которые были выпущены и проверены центром сертификации в

рамках инфраструктуры открытого ключа для аутентификации личности при обмене информацией через Интернет.

Факторы аутентификации

Аутентификация пользователя с идентификатором пользователя и паролем обычно считается самым основным типом аутентификации, и это зависит от пользователя, знающего две части информации: идентификатор пользователя или имя пользователя и пароль. Поскольку этот тип аутентификации основан только на одном факторе аутентификации, это тип однофакторной аутентификации.

Фактор аутентификации представляет собой часть данных или атрибута, которые могут использоваться для аутентификации пользователя, запрашивающего доступ к системе. Ранее выделялись такие факторы аутентификации, как: «то, что вы знаете, то, что у вас есть или что-вы такое». Эти три фактора соответствуют *фактору знания, фактору владения и фактору сущности*. В последние годы были предложены и введены в действие дополнительные факторы: местоположение и время.

Используемые в настоящее время факторы *аутентификации* включают:

Фактор знания: «То, что вы знаете». *Фактором* знаний могут быть любые учетные данные аутентификации, которые состоят из информации, которой обладает пользователь, включая персональный идентификационный номер (ПИН), имя пользователя, пароль или ответ на секретный вопрос.

Фактор владения: «То, что у вас есть». *Фактор* владения может представлять собой любые учетные данные, основанные на элементах, которыми пользователь может владеть и носить с собой, включая аппаратные устройства, такие как токен безопасности или мобильный телефон, используемый для приема текстового сообщения или запуска приложения аутентификации, которое может генерировать одноразоввый пароль или PIN-код.

Фактор сущности: «То, что вы есть». Фактор сущности обычно основан на некоторой форме биометрической идентификации, включая отпечатки пальцев или пальцев, распознавание лица, сканирование сетчатки или любую другую форму биометрических данных.

Фактор местоположения: «Где вы находитесь». Хотя он может быть менее конкретным, фактор местоположения иногда используется в качестве дополнения к другим факторам. Местоположение может быть определено с достаточной точностью устройствами, оснащенными GPS, или с меньшей точностью, путем проверки сетевых маршрутов. Фактор местоположения обычно не может использоваться для аутентификации сам по себе, но он может дополнять другие факторы, предоставляя средства для устранения некоторых запросов. Например, это может помешать злоумышленнику, находящемуся в удаленной географической области, создавать себя как пользователь, который обычно входит в систему только из дома или офиса организации в родной стране.

Фактор времени: «Когда вы аутентифицируетесь». Как и фактор местоположения, фактор времени сам по себе недостаточен, но он может быть дополнительным механизмом для отсеивания злоумышленников, которые пытаются получить доступ к ресурсу в то время, когда этот ресурс недоступен для авторизованного пользователя. Он также может использоваться вместе с местоположением. Например, если пользователь был последний раз аутентифицирован в полдень в США, попытка пройти аутентификацию из Азии через час будет отклонена на основе комбинации времени и местоположения.

Местоположение пользователя и текущее время сами по себе недостаточны для аутентификации пользователя без хотя бы одного из первых трех факторов. Однако вездесущность смартфонов помогает облегчить нагрузку многофакторной аутентификации для многих пользователей. Большинство смартфонов оснащены GPS, что обеспечивает разумную уверенность в подтверждении местоположения входа; MAC-адреса смартфона также могут использоваться для аутентификации удаленного пользователя, несмотря на то, что MAC-адреса относительно легко обманывать.

Аутентификация и авторизация

Авторизация включает процесс, посредством которого администратор предоставляет права на аутентифицированных пользователей, а также процесс проверки разрешений учетной записи пользователя, чтобы убедиться, что

пользователю предоставлен доступ к этим ресурсам. Права и привилегии, предоставленные для авторизированной учетной записи, зависят от разрешений пользователя, которые либо хранятся локально, либо на сервере аутентификации. Параметры, определенные для всех этих переменных среды, устанавливаются администратором.

Системе и процессам также может потребоваться авторизация их автоматизированных действий в сети. Онлайн-службы резервного копирования, системы исправления и обновления и системы удаленного мониторинга, такие как те, которые используются в технологиях телемедицины и смарт-сетки, все должны надежно пройти аутентификацию.

Типы методов аутентификации

Традиционная проверка подлинности зависит от использования файла паролей, в котором идентификаторы пользователей хранятся вместе с хэшами паролей, связанных с каждым пользователем. При входе в систему пароль, отправленный пользователем, хэшируется и сравнивается со значением в файле пароля. Если совпадают два хэша, пользователь аутентифицируется.

Такой подход к аутентификации имеет несколько недостатков, особенно для ресурсов, развернутых в разных системах. Во-первых, злоумышленники, которые могут получить доступ к файлу паролей для системы, могут использовать атаки грубой силы против хэшированных паролей для извлечения паролей. Для другого, этот подход потребует нескольких аутентификации для современных приложений, которые получают доступ к ресурсам в нескольких системах.

Недостатки аутентификации на основе паролей могут быть в определенной степени устранены с помощью более интеллектуальных имен пользователей и правил пароля, таких как минимальная длина и условия сложности, например, включая капиталы и символы. Однако аутентификация на основе пароля и аутентификация на основе знаний более уязвимы, чем системы, требующие нескольких независимых методов.

Другие методы проверки подлинности включают:

Строгая аутентификация – это термин, который формально не определен, но обычно используется для обозначения того, что используемый тип аутентификации является более надежным и устойчивым к атаке.

К строгой *аутентификации* в настоящее время относят многофакторную аутентификацию и аутентификацию на основе запрос-ответ с использованием криптографических методов.

Двухфакторная и многофакторная аутентификация

Добавление нескольких факторов аутентификации в процесс аутентификации обычно повышает безопасность. *Строгая* аутентификация обычно относится к аутентификации, которая использует, по меньшей мере, два фактора разных типов. Различие важно; так как имя пользователя и пароль можно рассматривать как типы факторов знания, можно сказать, что базовая аутентификация по имени пользователя и паролю используют два фактора знания для аутентификации – однако это не будет рассматриваться как форма двухфакторной аутентификации (2FA).

Двухфакторная аутентификация обычно зависит от фактора знания в сочетании с биометрическим фактором или фактором владения, таким как токен безопасности. Системы 2FA часто требуют, чтобы пользователь вводил код подтверждения, полученный посредством текстового сообщения на предварительно зарегистрированном мобильном телефоне, или код, созданный приложением аутентификации.

Многофакторная аутентификация требует от пользователей аутентификации с несколькими факторами аутентификации, включая биометрический фактор, такой как отпечаток пальца или распознавание лица, фактор владения, такой как фейс-ключ безопасности или токен, созданный приложением-аутентификатором.

Многофакторная аутентификация может включать аутентификацию любого типа, которая зависит от двух или более факторов, но процесс аутентификации по паролю плюс два разных типа биометрических данных, не будет считаться

трехфакторной аутентификацией, хотя, если для этого процесса необходим фактор знания, фактора владения и фактора сущности, тогда она будет трехфакторной.

Трехфакторная аутентификация (3FA) – это тип MFA, который использует три фактора аутентификации разных типов, как правило, коэффициент знания (пароль) в сочетании с фактором владения (маркер безопасности) и коэффициентом наследования (биометрический).

Примерами четырехфакторной аутентификации являются системы, которые требуют этих трех факторов, а также географический или временный фактор.

Одноразовый пароль. Одноразовый пароль представляет собой автоматически создаваемую числовую или буквенно-цифровую строку символов, которая аутентифицирует пользователя. Этот пароль действителен только для одного сеанса входа или транзакции и обычно используется для новых пользователей или для пользователей, потерявших свои пароли, и им предоставляется одноразовый пароль для входа в систему и перехода на новый пароль.

Биометрия. Хотя некоторые системы аутентификации могут зависеть исключительно от биометрической идентификации, биометрические данные обычно используются в качестве второго или третьего фактора аутентификации. Более распространенные типы биометрической аутентификации включают сканирование отпечатков пальцев, сканирование на лице или сетчатку и распознавание голоса.

Мобильная аутентификация. Мобильная аутентификация – это процесс проверки пользователя через их устройства или проверка самих устройств. Это позволяет пользователям входить в безопасные места и ресурсы из любого места. Процесс мобильной аутентификации включает в себя многофакторную аутентификацию, которая может включать одноразовые пароли, биометрическую аутентификацию или проверку QR-кода.

Непрерывная аутентификация. При непрерывной аутентификации, вместо того, чтобы пользователь вошел в систему или вышел из нее, приложение компании постоянно вычисляет «оценку подлинности», которая измеряет, насколько уверен,

что владелец учетной записи является физическим лицом, использующим устройство.

API (application programming interface) аутентификация. Стандартными методами управления API аутентификацией являются: базовая аутентификация HTTP; ключи API и OAuth.

В базовой аутентификации HTTP сервер запрашивает информацию аутентификации, то есть имя пользователя и пароль, от клиента. Затем клиент передает информацию аутентификации серверу в заголовке авторизации.

В методе аутентификации ключами API первому пользователю присваивается уникальное сгенерированное значение, которое указывает, что пользователь известен. Затем каждый раз, когда пользователь пытается снова войти в систему, его уникальный ключ используется для проверки того, что он тот же пользователь, который ранее вошел в систему.

Open Authorization (OAuth) является открытым стандартом для аутентификации и авторизации на токенах в Интернете. OAuth позволяет использовать информацию учетной записи пользователя сторонними службами, такими как Facebook, без раскрытия пароля пользователя. OAuth выступает в качестве посредника от имени пользователя, предоставляя службе токен доступа, который разрешает совместное использование определенной информации учетной записи.

Аутентификация пользователя и аутентификация компьютера

Идентификация машины может выполняться с учетными данными машины, как и идентификатор пользователя и пароль, только представленные данным устройством. Они также могут использовать цифровые сертификаты, выпущенные и проверенные центром сертификации, как часть инфраструктуры открытого ключа для подтверждения идентификации при обмене информацией через Интернет, например, тип цифрового пароля.

С увеличением числа устройств с поддержкой Интернета надежная аутентификация машины имеет решающее значение для обеспечения безопасной связи для автоматизированной системы «Умный дом» и другими приложениями

Интернета вещей, где практически любой субъект или объект могут быть адресатами и способны обмениваться данными по сети. Важно понимать, что каждая точка доступа является потенциальной точкой вторжения. Каждому сетевому устройству требуется надежная аутентификация машины, а также, несмотря на их обычно ограниченную активность, эти устройства также должны быть настроены для ограниченного доступа к разрешения, чтобы ограничить действия нарушителей.

3.2 Строгая аутентификация

Строгая аутентификация – это любой метод проверки подлинности пользователя или устройства, который по своей сути является достаточно строгим, чтобы обеспечить безопасность системы, которую она защищает, выдержав любые атаки, с которыми она может столкнуться.

Строгая аутентификация – это обычно используемый термин, который не имеет единого стандартизованного определения.

По мнению Европейского центрального банка (и многих организаций, которые следуют своим рекомендациям), строгая аутентификация объединяет как минимум два взаимозависимых фактора, так что компромисс одного метода не должен приводить к компромиссу второго. Кроме того, метод аутентификации должен включать один элемент, не подлежащий повторному использованию, который нельзя легко воспроизвести или украсть из Интернета. Т.е. в этом случае, термин строгая аутентификация является синонимом двухфакторной аутентификации или многофакторной аутентификации. Однако такая трактовка вводит в заблуждение, поскольку некоторые типы очень надежной аутентификации основаны на одном факторе аутентификации.

В некоторых случаях, например, в Recommendation X.509 считаются строгой аутентификацией системы, использующие множественные криптографически защищенные ответы на запрос/ответ. Однако такие системы основаны на множественных экземплярах фактора знания, а не на нескольких независимых

факторах и, как таковая, являются однофакторной аутентификацией (SFA). Аналогичным образом, некоторые типы биометрической аутентификации достаточно надежны при их использовании.

В национальном стандарте ГОСТ Р ИСО/МЭК 9594-8-98, аналогично международному стандарту Recommendation X.509 (11/88): The Directory – Authentication framework, выделяют два *вида аутентификации*: *простую*, основанную на простых паролях и *строгую*, осуществляемую удостоверениями личности, созданными с использованием криптографических методов [7]. В этом же стандарте выделяют и описывают по шагам три *процедуры строгой аутентификации*: однонаправленная, двунаправленная и трехнаправленная аутентификация.

В строгой аутентификации проверяемая (доказывающая) сторона доказывает свою подлинность проверяющей стороне, демонстрируя знание какого-либо секрета с помощью последовательности запросов и ответов с использованием криптографических методов и средств [37]. Секрет, как правило это секретный ключ, может быть предварительно распределен безопасным способом и в ходе аутентификационного обмена не пересылается по открытым каналам.

В односторонней аутентификации подлинность доказывает только одна сторона.

Двусторонняя аутентификация по сравнению с односторонней содержит аналогичный ответ проверяющей стороны доказывающей стороне, подтверждающий её подлинность.

Трехсторонняя аутентификация содержит дополнительную передачу данных от доказывающей стороны проверяющей, позволяет отказаться от использования меток времени при проведении аутентификации [38].

Протоколы строгой аутентификации могут быть основаны на использовании следующих криптографических методов:

- симметричных методов шифрования;
- однонаправленных ключевых хэш-функций;
- асимметричных методов шифрования;

– электронной цифровой подписи.

Двусторонняя и трехсторонняя процедуры относятся к *взаимной аутентификации*.

Взаимная аутентификация

Взаимная аутентификация или двусторонняя аутентификация относится к двум сторонам, аутентифицирующим друг друга одновременно, будучи режимом аутентификации по умолчанию в некоторых протоколах (IKE, SSH) и необязательным в других (TLS).

По умолчанию протокол TLS подтверждает только подлинность сервера для клиента с использованием сертификата X.509, а аутентификация клиента для сервера остается на уровне приложений. TLS также предлагает клиент-серверную аутентификацию с использованием аутентификации X.509 на стороне клиента. Поскольку это требует предоставления сертификатов клиентам и требует меньшего удобства для пользователя, он редко используется в приложениях для конечных пользователей.

Взаимная аутентификация TLS (mTLS) гораздо более широко распространена в приложениях для бизнеса (B2B), где ограниченное число программных и однородных клиентов подключаются к определенным веб-службам, эксплуатационная нагрузка ограничена, а требования к безопасности обычно намного выше, чем по сравнению с потребительской средой.

Лучшая аутентификация между учреждениями и клиентами помешала бы злоумышленникам успешно выдавать себя за финансовые учреждения для кражи учетных данных клиентов; и лучшая аутентификация между клиентами и учреждениями помешала бы злоумышленникам успешно выдавать себя за клиентов за финансовые учреждения для совершения мошенничества.

В большинстве случаев взаимная проверка подлинности выполняется по принципу «машина-машина», предоставляя возможность пользователям следить за случаями, когда удаленная проверка подлинности завершится неудачей (например, красный замок в адресной строке браузера или неправильное имя домена). Также существует для смягчения этой проблемы *нетехническая взаимная*

аутентификация, требующая от пользователя выполнения задачи, эффективного принуждения его к уведомлению и блокированию аутентификации с использованием ложной конечной точки.

Взаимная аутентификация бывает двух типов [46]:

- на основе сертификата;
- на основе имени и пароля пользователя.

При взаимной аутентификации сервер и клиент аутентифицируют друг друга. При использовании взаимной аутентификации на основе сертификатов выполняются следующие действия.

1 Клиент запрашивает доступ к защищенному ресурсу.

2 Веб-сервер представляет свой сертификат клиенту.

3 Клиент проверяет сертификат сервера.

4 В случае успеха клиент отправляет свой сертификат на сервер.

5 Сервер проверяет учетные данные клиента.

6 В случае успеха сервер предоставляет доступ к защищенному ресурсу, запрошенному клиентом.

На рисунке 3.1 показана схема шагов взаимной аутентификации на основе сертификатов.

При взаимной аутентификации на основе имени пользователя и пароля выполняются следующие действия.

1 Клиент запрашивает доступ к защищенному ресурсу.

2 Веб-сервер представляет свой сертификат клиенту.

3 Клиент проверяет сертификат сервера.

4 В случае успеха клиент отправляет свое имя пользователя и пароль на сервер, который проверяет учетные данные клиента.

5 Если проверка прошла успешно, сервер предоставляет доступ к защищенному ресурсу, запрошенному клиентом.

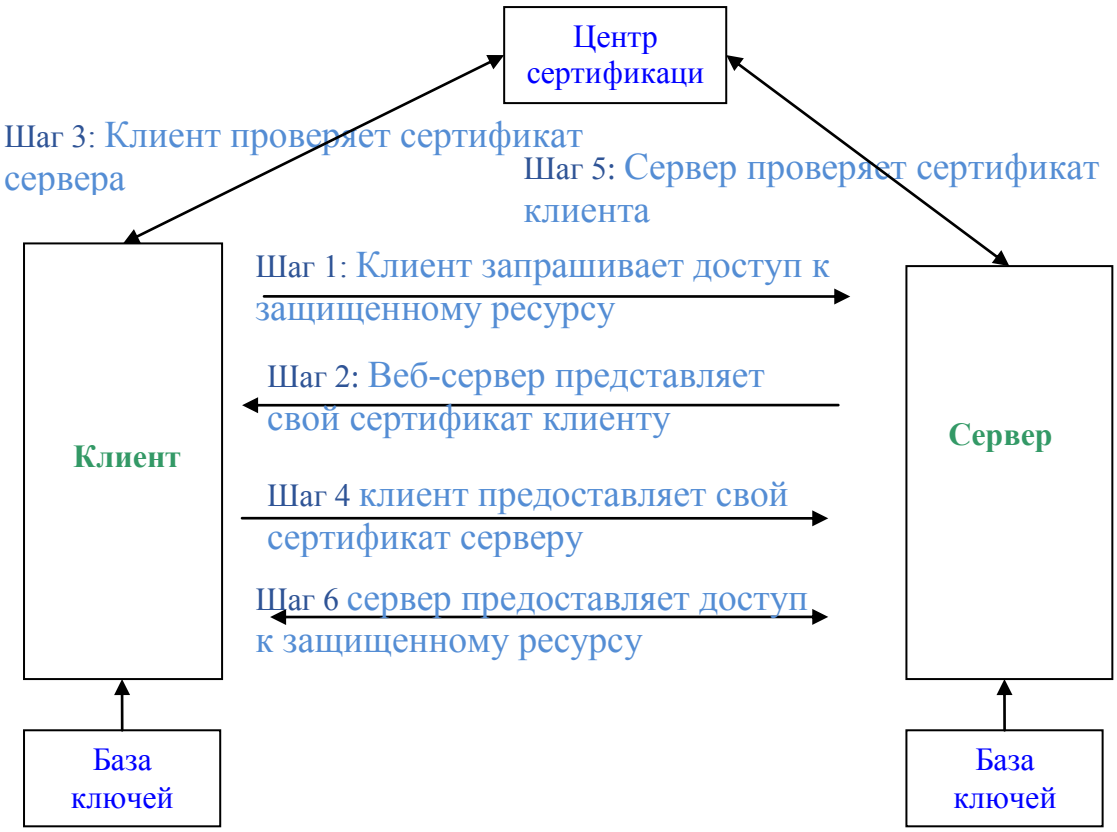


Рисунок 3.1 Взаимная аутентификация на основе сертификатов

На рисунке 3.2 показана схема взаимной аутентификации на основе имени пользователя и пароля.

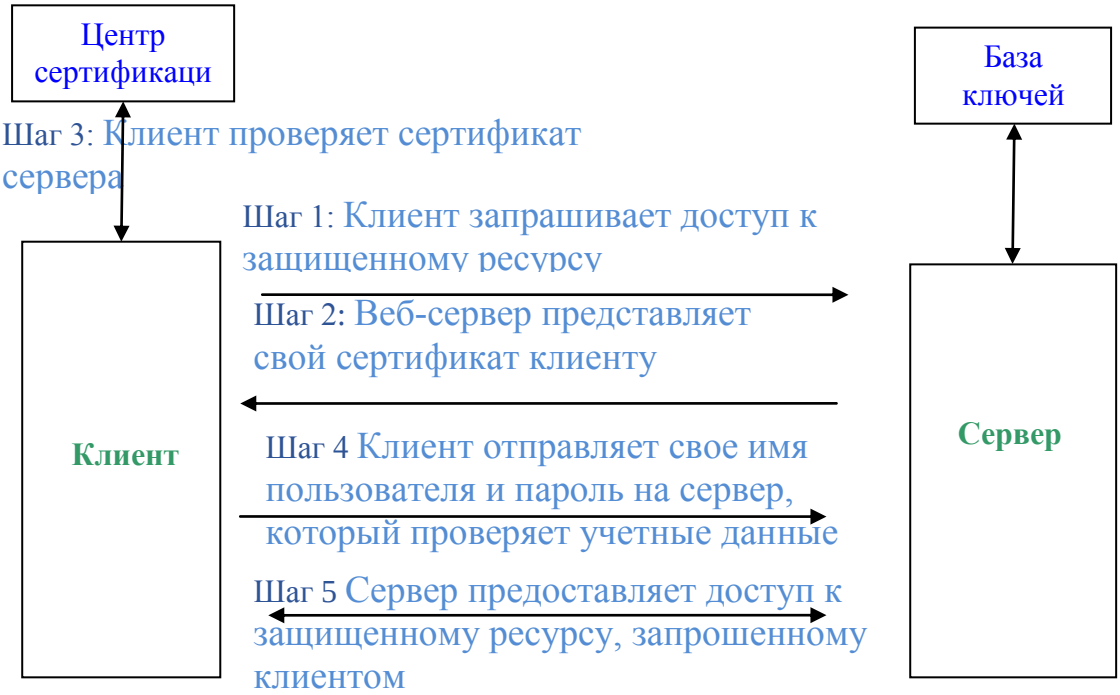


Рисунок 3.2 – Взаимная аутентификация на основе пароля и имени пользователя

3.2.1. Протоколы аутентификации с симметричными алгоритмами шифрования

Представителями централизованных протоколов аутентификации являются протоколы распределения секретных ключей: Нидхэма & Шредера и Kerberos.

Далее будут рассмотрены следующие варианты аутентификации:

- односторонняя аутентификация с использованием меток времени или с использованием случайных чисел;
- двусторонняя аутентификация.

Знание секретного ключа доказывается путем расшифрования запросов с его помощью.

Используют обозначения:

r_A – случайное число, сгенерированное пользователем *A*;

r_B – случайное число, сгенерированное пользователем *B*;

t_A – метка времени, сгенерированная пользователем *A*;

E_K – симметричное шифрование на ключе *K* (ключ *K* должен быть предварительно распределен между *A* и *B*).

A → B: M – Пользователь A отправляет пользователю B сообщение M.

1. Односторонняя аутентификация, основанная на метках времени:

$$A \to B : E_K(t_A, B)$$

Пользователь *B* проверяет правильность метки времени t_A и идентификатора *B*, которые нельзя изменить без знания секретного ключа. Это защищает от повторной передачи сообщения.

2. Односторонняя аутентификация, основанная на использовании случайных чисел:

$$B \to A : r_B$$

$$A \to B : E_K(r_B, B)$$

Пользователь *A* шифрует сообщение, состоящее из полученного случайного числа r_B и идентификатора *B* секретным ключом *K*, и отправляет его пользователю *B*. Только *B,* владеющий секретным ключом сможет расшифровать его и проверить правильность r_B.

3. Двусторонняя аутентификация, использующая случайные значения:

$$B \to A : r_B$$

$$A \to B : E_K(r_A, r_B, B)$$

$$B \to A : E_K(r_A, r_B)$$

Аналогично односторонней аутентификации оба пользователя подтверждают свою подлинность путем шифрования их общим секретным ключом *K* случайных чисел.

3.2.2. Протоколы, основанные на использовании однонаправленных ключевых хэш-функций

При аутентификации с помощью односторонней хэш-функции все пользователи сеанса используют одну и ту же процедуру одностороннего шифрования. Выделяют два способа.

Первый способ. Односторонняя хэш-функция h_K с параметром-ключом *K*, примененная к сообщению M произвольного размера, возвращает хэш-значение *m* (дайджест) фиксированного размера:

$$m = h_K(M).$$

1 Пользователь A вычисляет $m = h_K(M)$ и отправляет сообщение:

$$A \to B : m, M$$

2 Пользователь B, зная h_K и ключ *K* вычисляет свой дайджест $m' = h_K(M)$ и сравнивает с полученным дайджестом: $m' = m?$, при совпадении делает вывод, что сообщение не изменено.

Знание дайджеста не дает возможности восстановить исходное сообщение, но позволяет проверить целостность данных.

Второй способ. Односторонняя хэш-функция *h(.)* без параметра-ключа, применяется к сообщению М, дополненному секретным ключом *К*.

1 Пользователь А вычисляет $m = h(M, K)$ и отправляет сообщение:

$$A \to B : m, M$$

2 Пользователь В, зная h и ключ *К* вычисляет свой дайджест $m' = h(M, K)$ и сравнивает с полученным дайджестом: $m' = m?$, при совпадении делает вывод, что сообщение не изменено.

3.2.3 Аутентификация с использованием асимметричных алгоритмов шифрования

Протокол аутентификации с использованием асимметричных алгоритмов

Обозначения:

r – случайное число;

$E_{K_A}(*)$ – алгоритм асимметричного шифрования, открытым ключом пользователя А;

K_A – открытый ключ пользователя А;

h (*) – хэш-функция.

Схема протокола аутентификации:

$$B \to A : h(r), B, E_{K_A}(r, B)$$

$$A \to B : r$$

Пользователь *B* демонстрирует знание *r* без раскрытия самого значения *r*

Пользователь А проверяет корректность r, B и *h*(*r*) и отправляет обратно *r*.

Пользователь *B* проверяет корректность полученного r, сравнивая с отправленным.

Схема протокола аутентификации – *модифицированный протокол Нидхэма и Шредера*:

$$A \to B : E_{K_B}(r_1, A)$$

$$B \to A : E_{K_A}(r_1, r_2)$$

$$A \to B : r_2$$

3.2.4 Аутентификация, основанная на использовании цифровой подписи

Рассматриваются протоколы аутентификация с использованием цифровой подписи, меток времени и случайных чисел представленные в рекомендациях X.509 и соответствующем национальном стандарте [7].

Вводятся обозначения:

t_A, r_A, r_B –временная метка и случайные числа соответственно;

S_A – подпись, сгенерированная пользователем *A*;

S_B – подпись, сгенерированная пользователем *B*;

$cert_A$ –сертификат открытого ключа пользователя *A*;

$cert_B$ –сертификат открытого ключа пользователя *B*.

Здесь сертификаты могут использоваться для подтверждения подлинности открытых ключей.

1. Односторонняя аутентификация с применением меток времени:

$$A \to B : cert_A, t_A, B, S_A(t_A, B)$$

Пользователь А проверяет корректность t_A, *B* и, используя открытый ключ из сертификата $cert_A$, корректность $S_A(t_A, B)$.

2. Односторонняя аутентификация с использованием случайных чисел:

$$B \to A : r_B$$

$$A \to B : cert_A, r_A, B, S_A(r_A, r_B, B)$$

Пользователь В проверяет корректность B и, используя открытый ключ из сертификата $cert_A$, корректность $S_A(r_A, r_{\mathbf{B}}, B)$. Подписанное случайное число r_A используется для предотвращения атак с выборкой открытого текста.

3. Двусторонняя аутентификация с использованием случайных чисел:

$$B \to A : r_B$$

$$A \to B : cert_A, r_A, B, S_A(r_A, r_B, B)$$

$$B \to A : cert_B, A, S_B(r_A, r_B, A)$$

В данном протоколе обработка первого и второго сообщений выполняется так же, как и в предыдущем протоколе, а третье сообщение обрабатывается аналогично второму сообщению.

3.3 Реализация протоколов аутентификации

Когда пользователь входит в систему, система должна аутентифицировать его (а иногда пользователю требуется аутентификация системы). Существует много протоколов аутентификации. Ниже кратко описаны некоторые из наиболее распространенных:

1) PAP: протокол аутентификации пароля - это простейшая форма аутентификации и наименее безопасная. Имена пользователей и пароли отправляются в незашифрованном виде в виде простого текста. Это, очевидно, очень старый метод, который больше не используется.

2) SPAP: Shiva Password Authentication Protocol протокол аутентификации Shiva – это расширение для PAP, которое шифрует имя пользователя и пароль, которые отправляются через Интернет.

3) CHAP: Challenge Handshake Authentication Protocol: Протокол проверки подлинности при вызове рукопожатия вычисляет хэш после входа пользователя в систему. Затем он разделяет этот хеш с клиентской системой. Периодически сервер будет запрашивать у клиента этот хэш. (Это часть задачи.) Если клиент не подвеждает хеш, тогда ясно, что сообщения были скомпрометированы.

4) MS-CHAP является расширением Microsoft для CHAP. Шаги в основном таковы:

1 После того, как фаза рукопожатия завершена, аутентификатор (часто сервер) отправляет одноранговое сообщение «вызов».

2 Аналоговый ответ отвечает значением, рассчитанным с использованием функции одностороннего хеша.

3 Аутентификатор проверяет ответ со своим собственным расчетом ожидаемого значения хэш-функции. Если значения совпадают, аутентификация подтверждается; в противном случае соединение должно быть прекращено.

4 В случайные интервалы аутентификатор отправляет новый вызов партнеру и повторяет шаги с 1 по 3. Целая цель CHAP заключается не только в аутентификации, но и в периодической повторной аутентификации, что предотвращает атаки на захват сеанса.

5) EAP: структура, часто используемая в беспроводных сетях и соединения точка-точка. Он был первоначально определен в RFC 3748, но с тех пор обновлен. Он обрабатывает передачу ключей и связанных с ними параметров. Существует несколько версий EAP. Он имеет множество вариаций, в том числе:

6) Протокол LEAP: облегченный расширяемый протокол аутентификации был разработан Cisco и широко используется в беспроводной связи. LEAP поддерживается многими операционными системами Microsoft, включая Windows 7 и более поздние версии. LEAP использует модифицированную версию MS-CHAP.

7) Расширяемый протокол аутентификации - Transport Layer Security безопасный транспортный уровень TLS для обеспечения аутентификации. Большинство реализаций EAP-TLS используют цифровые сертификаты X.509 для аутентификации пользователей.

8) Protected Extensible Authentication Protocol (PEAP): шифрует процесс аутентификации с помощью аутентифицированного туннеля TLS. PEAP был разработан консорциумом, включая Cisco, Microsoft и RSA Security. Он был впервые включен в Microsoft Windows XP.

9) Kerberos: Kerberos основан на протоколе Нихейма-Шредера, широко используется, особенно в операционных системах Microsoft. Он был изобретен в Массачусетском технологическом институте и получил свое название от мифической трехглавой собаки, которая, как полагали, охраняла ворота Ада. Система немного сложна, но основной процесс выглядит следующим образом: когда пользователь входит в систему, сервер аутентификации проверяет идентификатор пользователя, а затем связывается с сервером предоставления мандатов. (Они часто находятся на одной машине.) Сервер предоставления мандатов отправляет зашифрованный «мандат» на машину пользователя. Этот мандат идентифицирует пользователя при входе в систему. Позже, когда пользователю необходимо получить доступ к некоторому ресурсу в сети, машина пользователя использует этот мандат для выдачи мандатов для получения доступа к целевой машине. Для мандатов есть много проверок, и эти мандаты истекают в относительно короткие сроки. Поскольку Kerberos настолько широко используется, он рассматривается здесь более подробно, чем другие методы проверки подлинности. Хотя существуют варианты исполнения, основной процесс показан на рисунке 2.3.

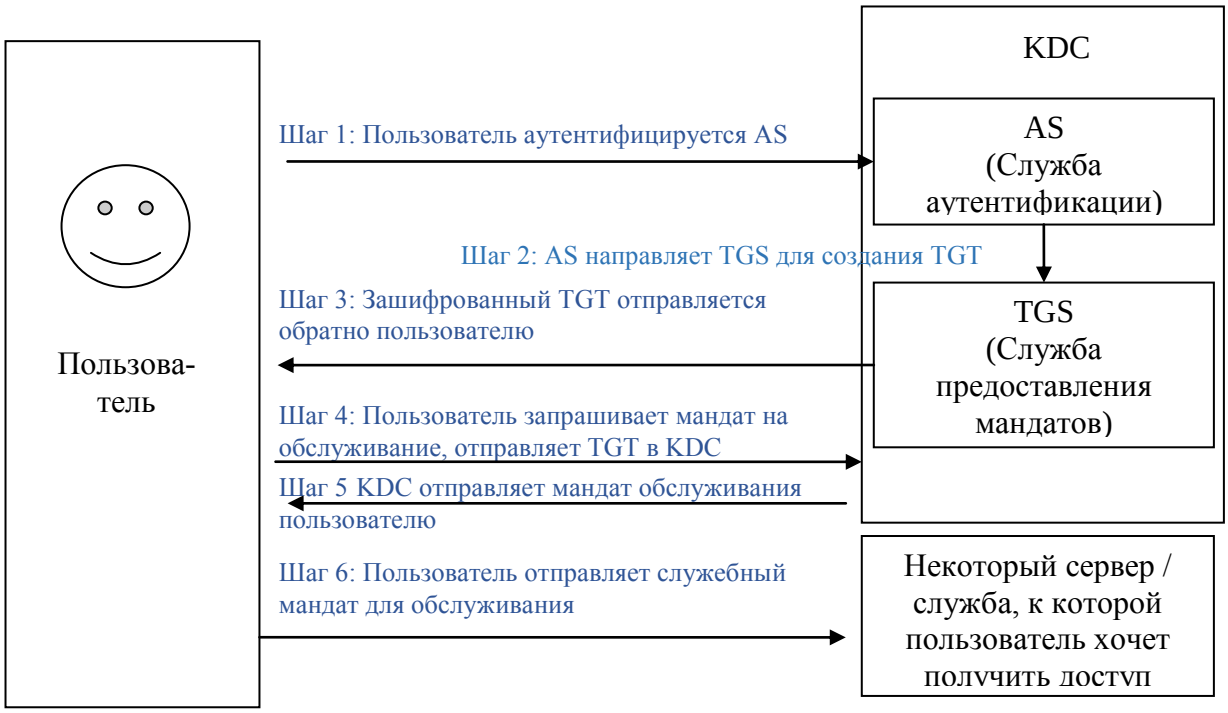


Рисунок 3.3 – Kerberos

Kerberos включает следующие элементы:

– Принципал: сервер или клиент, которым Kerberos может назначать мандаты.

– Сервер аутентификации (AS): сервер, который авторизует принципала и подключает его к серверу для подачи заявок.

– Сервер выдачи мандатов (TGS): Предоставляет мандаты.

– Центр распределения ключей (KDC): сервер, который предоставляет начальный мандат и обрабатывает запросы TGS. Часто он запускает службы AS и TGS.

Следует отметить, что Kerberos является одним из наиболее широко используемых протоколов аутентификации. Европа часто использует альтернативную SESAME Secure European System для приложений в среде с несколькими продуктами. Дополнительно описание Kerberos можно посмотреть в [40].

Таким образом, аутентификация приходит в разных видах и способах, и профессионалы в области безопасности должны знать как можно больше об аутентификации.

Контрольные вопросы

1 Какова роль аутентификации в кибербезопасности?

2 Что представляет собой фактор аутентификации?

3 Какие в настоящее время используются факторы аутентификации?

4 Какие существуют типы методов аутентификации

5 На использовании каких криптографических методов могут быть основаны протоколы строгой аутентификации

6 Какие методы аутентификации относят к строгой аутентификации?

7 Какие действия выполняются при использовании взаимной аутентификации на основе сертификатов?

8 Приведите схему протоколов аутентификации, основанных на использовании однонаправленных ключевых хэш-функций.

9 Приведите схему протоколов аутентификации, основанных на использовании асимметричных алгоритмов.

10 Приведите схему модифицированного протокола Нидхэма и Шредера для аутентификации.

11 Приведите схему протоколов аутентификации, основанных на использовании цифровой подписи.

12 Какие существуют реализации протоколов аутентификации?

13 Приведите основные шаги процесса работы протокола Kerberos.

14 Какие элементы включает протокол Kerberos?

4 Программно-аппаратные средства защиты информации в компьютерных системах

4.1 Сканеры вирусов

Сканер вирусов – это программное обеспечение для предотвращения заражения КС вирусом.

Антивирусные сканеры работают двумя основными способами [13, 21,43]. Первый заключается в том, что они содержат *список всех известных* определений *вирусов*. *Определение вирусов* – это простые файлы, в которых перечислены известные вирусы, их размер, свойства и поведение.

Как правило, одной из услуг, предоставляемых поставщиками антивирусных сканеров, является периодическое обновление этих файлов. Список описаний вирусов обычно находится в небольшом файле, который часто называется .dat-файлом (сокращение от data). При обновлении определений вирусов текущий файл заменяется более последним с веб-сайта поставщика. Затем антивирусная программа может сканировать компьютер, сеть и входящую почту для поиска известных вирусных файлов, т.е. поиск *сигнатуры* или *образца* (signature or pattern) известного вируса.

Любой файл на ПК или прикрепленный к электронному письму сравнивается с файлом определения вируса, чтобы узнать, есть ли совпадения. С помощью электронной почты это можно сделать, если искать конкретные темы и контент. Определения вирусов часто также содержат сведения о файле, размере файла и т. д. Это обеспечивает полную *сигнатуру* вируса.

Второй способ, работы антивируса, *поиск поведения*, характерного вирусам. По сути, сканер ищет действия, типичные для вирусов.

Сюда может входить любая программа, которая пытается записать в загрузочный сектор жесткого диска, сменить системные файлы, автоматизировать ваше почтовое программное обеспечение или саморазмножаться. Программы, которые пытаются изменить системный реестр (для систем Windows) или изменить

любые системные настройки, также могут указывать на наличие вируса. Очевидно, что этот второй метод – это, по сути, лучшее предположение.

Другая функция антивирусов поиск файлов, которые остаются в памяти после работы вируса. Это называется *Terminate и Stay Resident* (TSR) программой [43]. Некоторые законные программы делают это, но часто это признак вируса. Кроме того, некоторые антивирусные сканеры используют более сложные методы, такие как сканирование системных файлов и мониторинг любой программы, которая пытается изменить эти файлы.

Для любого поведения, антивирусное программное обеспечение использует определенные алгоритмы для оценки вероятности того, что данный файл фактически является вирусом. Следует отметить, что современные антивирусные сканеры просматривают все виды вредоносных программ, включая троянские программы, шпионские программы и вирусы.

Существует третий метод, называемый *эвристическим сканированием, представляющий* в основном исследование самого файла и похож на метод сканирования сигнатур. Однако в этом случае файл не должен точно соответствовать сигнатуре.

Эвристика относится к функциям, которые оценивают различные альтернативы, используя шаг ветвления в алгоритме. Таким образом, эвристическое сканирование проверяет вероятность того, что данный файл является вирусом, основано на *характеристиках* файлов, а не на поведении. Различают сканирование по требованию и постоянное. Постоянный вирусный сканер работает в фоновом режиме и постоянно проверяет компьютер на наличие каких-либо признаков вируса.

Сканеры по требованию работают только при их запуске. Многие современные антивирусные сканеры предлагают оба варианта.

Имейте в виду, что любая антивирусная программа будет иметь некоторые *ложные срабатывания* и некоторые *пропуски*. Ложное срабатывание возникает, когда вирус-сканер определяет данный файл как вирус, а на самом деле это не так. Например, законная программа может редактировать ключ реестра или взаимодействовать с адресной книгой электронной почты. Пропуск возникает, когда

вирус ложно считается законной программой. Из-за ложных срабатываний рекомендуется, чтобы ваш антивирус автоматически не удалял подозрительные вирусы, а помещал на карантин, и уведомлял пользователя компьютера.

4.1.1 Методы сканирования вирусов

Выделяют у антивирусов пять способов обнаружения заражения вирусами [43]:

1) *Сканирование электронной почты и вложений*: поскольку основным методом распространения вируса является электронная почта, сканирование по электронной почте и вложения – самая важная функция любого антивирусного сканера.

Некоторые сканеры вирусов фактически проверяют электронную почту на сервере электронной почты перед загрузкой ее на компьютер получателя почты. Другие антивирусные сканеры работают, сканируя свои электронные письма и вложения на получающем компьютере, прежде чем передавать их в почтовую программу. В любом случае электронная почта и ее вложения должны быть отсканированы до того, как у вас появится возможность открыть их и выпустить вирус в систему.

2) *Сканирование загрузки*. В любое время, когда вы загружаете что-либо из Интернета, либо через веб-ссылку, либо через какую-либо FTP-программу, вы можете загрузить зараженный файл. Сканирование загрузки работает так же, как сканирование электронной почты и вложений, но делает это в файлах, выбранных для загрузки.

3) *Сканирование файлов*. Это тип сканирования, в котором проверяются файлы в системе, чтобы убедиться, что они соответствуют любому известному вирусу. Такое сканирование обычно выполняется по требованию или периодически, а не на постоянной основе.

4) *Эвристическое сканирование*. Возможно, самая передовая форма сканирования вирусов, использует правила, чтобы определить, ведет ли файл или

программу как вирус и является одним из лучших способов найти вирус, который не является известным вирусом. Новый вирус не будет включен в список определений вирусов, поэтому вы должны изучить его поведение, чтобы определить, является ли он вирусом. Однако этот процесс не является надежным. Некоторые фактические вирусные инфекции будут пропущены, а некоторые файлы без вирусов могут быть заподозрены в вирусе.

5) *Песочница*: используется отдельная область, изолированная от операционной системы, в которой выполняется загрузка или вложение. Если файл заражен, он не сможет заразить операционную систему.

Одним из способов выполнения песочницы является то, что операционная система выделяет защищенную область памяти, чтобы открыть подозрительный файл и контролировать его поведение. Это не на 100% эффективно, но гораздо безопаснее, чем просто открывать файлы в системе и надеяться, что заражения не произойдет. Способ полезно применять в корпоративных сетях. На выделенной рабочей станции настраивается система, идентичная конфигурации на стандартных рабочих станциях. Тем не менее, этот рабочая станция не подключена к сети. Сначала в этой системе открываются подозрительные файлы. Затем система отслеживается на предмет наличия признаков инфекции. После проверки файла, его можно будет открыть на обычных рабочих станциях. Простым способом сделать это в домашнем или малом офисе является создание виртуальной машины на компьютере, на которой сначала и открываются подозрительные вложения или загрузки. На этой виртуальной машине могут быть установлены антивирусные сканеры. Кроме того, можно изменить время на виртуальной машине, чтобы обнаружить логические бомбы. Оставить подозрительный файл находиться на виртуальной машине в течение определенного периода времени, прежде чем перенести его на главный компьютер.

Как работают большинство коммерческих сканеров.

Большинство коммерческих антивирусных сканеров используют несколько методов, включая большинство перечисленных выше методов. Любой вирусный сканер, который использует только один способ сканирования, практически

бесполезен для практической защиты от вирусов. Использование этих методов характеризуются тем, как работает сканер независимо от того, использует ли он эвристическое сканирование, сканирование загрузки, сканирование по электронной почте и т. д.

– *Сканирование активного кода*. Современные веб-сайты часто включают активные коды, такие как Java-апплеты и ActiveX. Эти технологии могут обеспечить некоторые интересные визуальные эффекты для любого веб-сайта. Тем не менее, они также могут быть средством для вредоносного кода. Сканирование таких объектов перед их загрузкой на компьютер является важной особенностью любого качественного антивируса.

– *Ложные срабатывания и пропуски*. Независимо от типа антивирусного сканера, любое антивирусное программное обеспечение будет иногда ошибаться. Существует два типа ошибок, которые нужно учитывать. – Антивирусное программное обеспечение может принять законную программу за вирус. Например, у вас может быть программа, которая должна внести некоторую корректировку в реестр Windows или для сканирования вашей адресной книги электронной почты. Ошибки принятия законной программы за вирус называют ложным срабатыванием. Также антивирус может не распознать вирус, тогда говорят о пропуске. Лучший способ минимизировать пропуски – выполнять регулярно обновление антивирусного программного обеспечения. Для ложных срабатываний рекомендуется просто помещать в карантин подозрительные вирусы и не удалять их автоматически.

Коммерческое антивирусное программное обеспечение

Существует несколько брендов антивирусного программного обеспечения, которые сегодня фактически доминируют на антивирусном рынке, а ряд компаний, предлагающих коммерческий сканер, также предлагают бесплатную версию, которая не предоставляет столько функций, как коммерческий продукт. Например, AVG AntiVirus, доступный на www.grisoft.com, является коммерческим продуктом, но компания также предлагает AVG AntiVirus Free Edition. McAfee, Norton и Kapersky - три наиболее популярных производителя антивирусов. Все три продукта

являются хорошим выбором и оснащены дополнительными опциями, такими как спам-фильтры и персональные брандмауэры. Любой из этих продуктов можно приобрести для домашней машины по цене от $ 30 до $ 60 (в зависимости от добавляемых опций). Эта цена покупки включает годовую подписку для обновления вирусных файлов, чтобы ваше антивирусное программное обеспечение распознало все известные вирусные атаки. Организационные лицензии также доступны для охвата всей сети [43].

Malwarebytes – еще один популярный продукт, который имеет как бесплатные, так и коммерческие версии. Есть и другие антивирусные решения, которые можно легко найти в Интернете.

4.1.2 Анализ вредоносных программ

Анализ вредоносных программ предполагает изучение функциональности, цели, происхождения и потенциального влияния вредоносного программного обеспечения. Эта задача традиционно очень сложная и трудоемкая, требующая аналитиков с экспертными знаниями в области внутреннего программного обеспечения и реверс-инжиниринга [13, 44].

Наука о данных и машинное обучение показали многообещающую способность автоматизировать некоторые части анализа вредоносных программ, но эти методы все еще в значительной степени зависят от извлечения значимых признаков из данных, что является нетривиальной задачей, которая по-прежнему требует практиков со специализированными навыками.

Понимание вредоносного ПО

Исходный код проходит ряд этапов, прежде чем запускаться в качестве программы на компьютере. Понимание этих шагов имеет решающее значение для любого аналитика вредоносных программ. Существует примерно столько же различных типов вредоносных программ, сколько существует различных типов программного обеспечения, причем каждый тип потенциально написан на своем языке программирования, ориентирован на разные среды выполнения и имеет

разные требования к исполнению. Имея доступ к высокоуровневому коду (например, C / C ++, Java или Python), относительно легко понять, что делает программа и как профилировать ее поведение. Однако маловероятно получить легкий доступ к высокоуровневому коду, используемому для создания вредоносных программ.

В своем упакованном и развернутом состоянии большинство вредоносных программ существует в виде двоичных файлов, которые часто не читаются человеком и предназначены для непосредственного выполнения на компьютере. Профилирование характеристик и поведения вредоносного ПО затем превращается в процесс обратного инжиниринга, чтобы выяснить, что оно делает, как если бы у нас был доступ к высокоуровневому коду.

Двоичные файлы по своей природе запутаны, что создает большие трудности для тех, кто пытается получить информацию из них. Без знания контекста, стандартов кодирования и алгоритма декодирования сами двоичные данные не имеют смысла.

Например, если нужно классифицировать музыкальные файлы WAV3 в разные музыкальные жанры (например, классика, рок, поп, джаз), то необработанными данными будут WAV-файлами. Наиболее прямой перевод каждого WAV-файла на вход алгоритма анализа состоит в том, чтобы использовать двоичное представление файла на уровне битов. Однако это не самое эффективное представление музыкальных файлов. Вместо этого можно выполнить функциональное проектирование необработанного ввода для генерации других представлений этих данных.

Например, можно было бы запустить его через программу анализа музыки, чтобы извлечь характеристики, такие как минимальная, максимальная и средняя амплитуда и частота. Более сложные программы анализа могут быть в состоянии извлечь такие функции, как количество ударов в минуту, музыкальный ключ, в котором находится произведение, и более тонкие полифонические характеристики музыки. Эти характеристики могут помочь нарисовать гораздо более полную картину каждого музыкального произведения, позволяя классификатору машинного

обучения выявлять различия в темпе, ритме и тональных характеристиках между образцами разных жанров.

Для выявления и извлечения характеристик, полезных для выполнения анализа безопасности в двоичных файлах компьютера требуется глубокое понимание внутренних компонентов программного обеспечения. Эта область исследования называется обратным инжинирингом программного обеспечения - процессом извлечения информации и знаний о внутренней работе программного обеспечения для полного понимания его свойств, принципов работы и недостатков. Используя обратный инжиниринг двоичного файла, можно понять его функциональность, его назначение, а иногда даже его происхождение.

Вредоносное программное обеспечение может быть встроено во множество различных двоичных форматов, которые работают совсем не так, как другие. Например, файлы Windows PE (переносимые исполняемые файлы с расширениями .exe, .dll, .ei и т. Д.), Файлы Unix ELF (исполняемые файлы и формат с возможностью связывания) и файлы APK для Android (Android Формат Package Kit, с расширениями (.apk и т. д.), Имеет очень разную структуру файлов и контексты выполнения. Фон, необходимый для анализа каждого класса исполняемых файлов, также различен.

Также необходимо учитывать вредоносные программы, которые существуют в других формах, кроме отдельных исполняемых двоичных файлов. Вредоносное ПО основанное на документах, такими как .doc, .pdf и .rtf.

Макрос – набор команд для автоматизации определенных повторяющихся задач в контексте приложений, таких как Microsoft Word или Excel. Макросреда была широко распространена в 1990-х годах, используя возможности этих популярных программ для автоматизации выполнения вредоносного кода на компьютере жертвы. В последние годы вредоносное ПО для макросов было возвращено, часто благодаря кампаниям социальной инженерии, обеспечивающим широкое распространение.

Обычно используется макросы и динамические исполняемые элементы в структуре документа для выполнения злонамеренных действий. Вредоносные

программы также могут быть представлены в виде расширений и плагинов для популярных программных платформ, таких как веб-браузеры и веб-фреймворки.

4.1.3 Классификация вредоносных программ

Классификация вредоносных программ объединяет отдельные образцы вредоносных программ на основе общих свойств. Можно классифицировать вредоносные программы различными способами [13, 14, 21], в зависимости от цели задачи. Например, можно сгруппировать вредоносное ПО по степени серьезности и функциям, чтобы эффективно оценить риск, который оно представляет для организации. Также можно сгруппировать вредоносное ПО по потенциальным масштабам повреждения и векторам проникновения, чтобы разработать стратегии устранения и смягчения последствий.

Исследователи вредоносного ПО могут классифицировать вредоносное ПО по происхождению и авторству, чтобы понять его генеалогию и цель.

Для анализа вредоносных программ общего назначения в отрасли следует группировать выборки по семейству – термин, используемый аналитиками вредоносных программ, который позволяет отслеживать авторство, сопоставлять информацию и выявлять новые варианты недавно обнаруженных вредоносных программ. Образцы вредоносных программ одного и того же семейства могут иметь сходный код, возможности, авторство, функции, цели, и / или происхождение.

Между вредоносным ПО существует *метаморфизм* и *полиморфизм*. Полиморфное вредоносное ПО обычно состоит *из двух частей*: основной логики, которая выполняет заражение, и другого охватывающего раздела, который использует различные формы шифрования и дешифрования, чтобы скрыть код заражения.

Метаморфное вредоносное ПО внедряет, изменяет переопределяет, добавляет и удаляет код во вредоносном ПО. Поскольку логика заражения не изменяется между каждым этапом эволюции вредоносного ПО, сравнительно легче обнаружить полиморфное вредоносное ПО, чем метаморфное вредоносное ПО.

Классификация вредоносных программ также может быть обобщена, включением классификации вредоносных двоичных файлов. Этот тип классификации используется для определения того, является ли часть программного обеспечения вредоносной. Учитывая произвольный двоичный файл, можно узнать вероятность доверия этому файлу и выполнить его в доверенной среде. Это основная цель антивирусного программного обеспечения и является особенно важной задачей для специалистов по компьютерной безопасности, поскольку эти знания могут помочь предотвратить распространение вредоносного ПО в организации.

Традиционно эта задача обусловлена сопоставлением сигнатур: с учетом множества свойств и поведения ранее замеченного вредоносного ПО новые входящие двоичные файлы можно сравнивать с этим набором данных, чтобы определить, соответствует ли оно чему-либо увиденному ранее.

Метод сопоставления сигнатур работает хорошо до тех пор, пока авторам вредоносных программ не удается значительно изменить свойства и поведение вредоносных программ, чтобы избежать обнаружения, а выбранные свойства и поведение имеют хороший баланс стабильности сигнала (поэтому все Образцы вредоносного ПО, принадлежащие этому семейству, демонстрируют этот сигнал) и отличительные особенности (таким образом, доброкачественные двоичные файлы не будут демонстрировать свойства или поведение, которые позволяют их ошибочно классифицировать как вредоносное ПО).

Тем не менее, авторы вредоносных программ имеют сильный стимул постоянно изменять свойства и поведение своего программного обеспечения, чтобы избежать обнаружения.

Метаморфические или полиморфные вирусы и черви используют методы статической и динамической обфускации [13], чтобы изменить характеристики своего кода, поведения и свойств, используемые в алгоритмах генерации сигнатур механизмов идентификации вредоносных программ. Этот уровень сложности вредоносных программ сначала был редким, но стал более распространенным из-за продолжающегося успеха в борьбе механизмов сигнатур с вредоносными

программами. Механизмы сигнатур продолжают преследовать постоянно сужающийся набор статических сигналов, которые авторы вредоносных программ пренебрегают скрыть или принципиально не могут изменить.

4.2 Межсетевые экраны

Межсетевой экран или брандмауэр (или англ. firewall) – это, по сути, барьер между двумя компьютерами или компьютерными системами. Самое распространенное место расположения межсетевого экрана – это внутренняя сеть и выход во внешнюю сеть. Тем не менее, брандмауэры на отдельных компьютерах и между сегментами сети также довольно распространены [11,43].

Как минимум, брандмауэр будет фильтровать входящие пакеты на основе определенных параметров, таких как размер пакета, IP-адрес источника, протокол и порт назначения. Linux и Windows (начиная с Windows XP и во всех последующих версиях Windows) поставляются с простым брандмауэром. Брандмауэр в ОС Windows 7 был расширен для обработки фильтрации входящего и исходящего трафика. Windows 8 и Windows 10 существенно не изменили функциональность Windows-брандмауэра.

Следует включить и настроить индивидуальные компьютерные брандмауэры в дополнение к брандмауэрам периметра. В организационной настройке понадобится, как минимум, выделенный межсетевой экран между защищаемой сетью и внешним миром. Это может быть маршрутизатор, который также имеет встроенные возможности брандмауэра. (Cisco Systems – одна из компаний, которая хорошо известна высококачественными маршрутизаторами и брандмауэрами.) Или это может быть сервер, который предназначен исключительно для работы с программным обеспечением брандмауэра. Однако выбор брандмауэра – важное решение, требующее определенного опыта для принятия этого решения.

Преимущества и ограничения брандмауэров

Брандмауэр, независимо от типа, в основном является инструментом блокировки определенного трафика. Набор правил определяет, какой трафик

разрешен и какой трафик блокируется. Очевидно, что брандмауэр является важной частью стратегии безопасности КС [29, 31, 43].

Тем не менее, это не панацея для безопасности, поскольку она не может блокировать каждую атаку. Например, брандмауэр не остановит пользователя от загрузки троянского коня. Он также не может остановить внутренние атаки. Но брандмауэр может быть отличным способом остановить атаку отказа в обслуживании (DoS) или предотвратить сканирование внутренних деталей вашей сети хакером.

4.2.1 Типы и компоненты межсетевых экранов

Существует множество типов брандмауэров и вариантов этих типов. Но большинство брандмауэров можно сгруппировать в одно из следующих трех семейств брандмауэров [11, 31, 43].

- Пакетные фильтры (Packet inspection).
- Инспекторы состояния пакетов (Stateful packet inspection – SPI).
- Посредники прикладного уровня (Application Gateway).

Пакетные фильтры

Базовая фильтрация пакетов – это простейшая форма брандмауэра. Он рассматривает пакеты и проверяет каждый пакет на соответствие правилам брандмауэра. Например, для брандмауэра с фильтрацией пакетов часто задают три вопроса:

1 Этот пакет использует протокол, разрешенный брандмауэром?

2 Этот пакет предназначен для порта, разрешенного брандмауэром?

3 Является ли пакет исходящим из IP-адреса, который брандмауэр не заблокировал?

Это три основных *правила*. Некоторые брандмауэры с фильтром пакетов имеют дополнительные правила для проверки.

Но, предыдущие пакеты из того же источника не учитываются при проверке текущих. По сути, каждый пакет рассматривается как единичное событие без связи с

предыдущими. Это делает брандмауэры фильтрации пакетов весьма восприимчивыми к некоторым DoS-атакам, таким как SYN-floods.

Инспекторы состояния пакетов

Брандмауэр Malwarebytes будет проверять каждый пакет, отказывать или разрешать доступ, основываясь не только на рассмотрении текущего пакета, но также на данных, полученных из предыдущих пакетов в сеансе. Это означает, что брандмауэр знает контекст, в котором был отправлен определенный пакет.

Это делает эти брандмауэры гораздо менее восприимчивыми к флуд-атакам: ping floods и SYN floods, а также менее подверженным спуфингу (spoofing). Например, если брандмауэр обнаруживает, что текущий пакет является ICMP-пакетом, и поток из нескольких тысяч пакетов постоянно поступает с одного и того же исходного IP-адреса, это явно DoS-атака, и пакеты будут заблокированы.

Брандмауэр SPI также может просматривать фактическое содержимое пакета, что позволяет использовать некоторые очень продвинутые возможности фильтрации. Большинство брандмауэров класса high-end используют метод проверки пакетов с сохранением состояния; когда это возможно, это рекомендуемый тип брандмауэра.

Посредники прикладного уровня

Посредники прикладного уровня или шлюз приложения (также известный как прокси-сервер приложения или прокси-сервер уровня приложения) – это программа, работающая на брандмауэре. Когда клиентская программа, такая как веб-браузер, устанавливает соединение с целевой службой, например, веб-сервером, она подключается к шлюзу приложения или прокси. Затем клиент ведет переговоры с прокси-сервером, чтобы получить доступ к целевой службе. Фактически, прокси устанавливает соединение с получателем за брандмауэром и действует от имени клиента, скрывая и защищая отдельные компьютеры в сети за брандмауэром.

Этот процесс фактически создает *два соединения*. Между клиентом и прокси-сервером существует одно соединение и другое соединение между прокси-сервером и пунктом назначения. После установления соединения шлюз приложения принимает все решения о пересылке пакетов. Поскольку вся связь осуществляется

через прокси-сервер, компьютеры, расположенные за брандмауэром, защищены. По сути, брандмауэр приложения – это тот, который используется для определенных типов приложений, таких как база данных или веб-сервер. Он способен исследовать используемый протокол (например, HTTP) для любого аномального поведения и блокировать трафик, который может пройти мимо других типов брандмауэров. Обычно используется брандмауэр приложения, который также включает SPI.

4.2.2 Конфигурации межсетевых экранов

В дополнение к различным типам брандмауэров существуют различные *варианты конфигурации*. Тип брандмауэра дает представление о том, как будет оцениваться трафик и, следовательно, как принимаются решения о том, что разрешать и не разрешать.

Конфигурация дает представление о настройках брандмауэра по отношению к защищаемой сети. К основным видам конфигураций / реализаций для брандмауэров относят следующее:

- Хост – сеть.
- Двудомный хост.
- Брандмауэр на основе маршрутизатора.
- Межсетевой экран на основе экранированного шлюза.

Хост - сеть

Брандмауэр хост – сеть представляет собой программное решение, установленное на существующем компьютере с существующей операционной системой. Наиболее значительная проблема при использовании этого типа брандмауэра заключается в том, что независимо от того, насколько хорошее это решение для брандмауэра, оно зависит от базовой операционной системы. В такой ситуации абсолютно важно, чтобы машина, на которой размещен брандмауэр, имела упрощенную операционную систему.

Двудомный хост

Двудомный хост (*dual-homed host)* – это брандмауэр, работающий на сервере с по меньшей мере двумя сетевыми интерфейсами. Сервер работает как маршрутизатор между сетью и интерфейсами, к которым он присоединен. Для выполнения этой работы функция автоматической маршрутизации отключена, что означает, что IP-пакет из Интернета не перенаправляется непосредственно в сеть. Можно выбрать, какие пакеты маршрутизировать и как их маршрутизировать. Системы внутри и снаружи брандмауэра могут взаимодействовать с двудомным хостом, но не могут общаться напрямую друг с другом.

Брандмауэр на основе маршрутизатора

Можно реализовать брандмауэр на маршрутизаторе – экранирующий или фильтрующий маршрутизатор. В больших сетях с несколькими уровнями защиты это обычно первый уровень защиты. Экранирующий маршрутизатор позволяет блокировать или фильтровать пакеты путем анализа их адресов и портов. При поступлении пакета межсетевой экран определяет соответствие данного пакета и состояния соединения правилам политики. Если пакет разрешен, то передается по своему маршруту, иначе отклоняется или аннулируется.

Хотя на маршрутизаторе можно реализовать различные типы брандмауэров, наиболее *распространенным типом является фильтрация пакетов*.

Межсетевой экран на основе экранированного шлюза

Межсетевой экран на основе экранированного шлюза или экранированный хост представляет собой комбинацию брандмауэров. В этой конфигурации используется комбинация фильтрующего маршрутизатора и прикладного шлюза (хоста-бастиона), размещаемого со стороны внутренней сети. Прикладной шлюз реализуется на хосте и имеет только один сетевой интерфейс. Экранирующий маршрутизатор добавляет безопасность, позволяя запретить или разрешить определенный трафик с хоста-бастиона.

Коммерческие и бесплатные продукты брандмауэра

Существует множество коммерческих продуктов брандмауэра, многие поставщики программного обеспечения предлагают базовое решение для фильтрации пакетов. Крупные поставщики антивирусных программ часто

предлагают программное обеспечение брандмауэра в комплекте с их антивирусным программным обеспечением.

Другие компании, такие как Zone Labs, предлагают брандмауэр и программное обеспечение для обнаружения вторжений (IDS). Zone Labs, например, предлагает ZoneAlarm Security Suite, который предоставляет все инструменты для полной безопасности в Интернете. Крупнейшие производители маршрутизаторов и концентраторов, такие как Cisco Systems, также предлагают продукты брандмауэра. Насколько нужна безопасность, это сложный вопрос для ответа. Минимальная рекомендация состоит в том, чтобы иметь межсетевой экран / прокси-сервер для фильтрации пакетов между сетью предприятия и Интернетом.

Также доступно множество бесплатных брандмауэров. Zone Labs предлагает бесплатную версию брандмауэра ZoneAlarm. Comodo Firewall, доступный по адресу https://www.comodo.com/home/internet-security/, представляет собой продукт, предназначенный для домашнего или малого офиса. Как и продукт Zone Labs, он имеет как бесплатную версию, так и расширенную коммерческую версию, бесплатная версия является более старой версией программного обеспечения и не включает многие усовершенствования коммерческой версии.

4.2.3 Журналы межсетевых экранов

Брандмауэры также являются отличными инструментами при попытке выяснить, что произошло после инцидента.

Почти все брандмауэры, независимо от типа или реализации, могут регистрировать активность. Эти журналы могут предоставлять ценную информацию, позволяющую определить источника атаки, методы, используемые для атаки, и другие данные, которые могут помочь либо найти исполнителя атаки, либо, по крайней мере, предотвратить будущую атаку, используя те же методы. Любой системный администратор, ответственный за безопасность, должен постоянно проверять журналы брандмауэра.

4.3 Антишпион

Антишпион (Antispyware) сканирует компьютер в поиске на компьютере шпионского ПО. Это важный элемент программного обеспечения компьютерной безопасности, которое в свое время в значительной степени игнорировался. Даже сегодня мало людей серьезно относятся к шпионскому программному обеспечению или защищают его. Большинство антишпионских программ работает, проверяя вашу систему на наличие известных шпионских файлов, сопоставляя каждое приложение со списком известных программ-шпионов. Следует поддерживать услугу подписки, чтобы получать обновления в списке определений шпионских программ. Большинство современных антивирусных решений также проверяют наличие шпионских программ. Использование антишпионских программ столь же важно, как и антивирусное программное обеспечение, несоблюдение этого может привести к серьезным последствиям. Шпионское ПО является причиной утечки личных данных и конфиденциальных бизнес-данных и может стать средством целенаправленного промышленного шпионажа.

Кроме использования антишпионских программ или в сочетании с ними можно защититься с помощью настроек безопасности браузера. Шпионское ПО может проникнуть через вложения и загрузки в Интернете, загрузки различных «улучшений» в Интернете, таких как «скины» и «панели инструментов». Запрещение таких загрузок должно быть вопросом политики компании. Многие веб-сайты сегодня нуждаются в какой-то надстройке (дополнения add-ins), такой как Flash, чтобы нормально функционировать, следует разрешать только надстройки с доверенных, известных сайтов.

4.4 IDS

В последние несколько лет IDS стала использоваться более широко. По сути, IDS будет проверять все входящие и исходящие действия порта на компьютере / брандмауэре / системе и искать шаблоны, которые могут указывать на попытку

взлома [16, 23, 31, 43]. Например, если IDS обнаруживает, что серия ICMP-пакетов была отправлена последовательно на каждый порт, это, вероятно, указывает на то, что система сканируется программным обеспечением сетевого сканирования, таким как Cerberus. Поскольку сканирование часто является предварительным этапом к попытке нарушить безопасность системы, может быть очень важно знать, что кто-то выполняет подготовительные шаги для проникновения в систему.

Важно, чтобы было общее представление о том, как работают эти системы. Далее приведены широкие категории, в которых, как правило, рассматриваются системы IDS, а затем некоторые конкретные подходы к IDS. Хотя представленная информация не является всеобъемлющей, она затрагивает наиболее распространенную терминологию.

4.4.1 Классификация IDS

Существует несколько способов, с помощью которых системы IDS можно классифицировать. Наиболее распространенными категориями IDS являются:

− Пассивная IDS.

− Активная IDS (также называемая системой предотвращения вторжений Intrusion Prevention System или IPS).

Пассивная IDS

Пассивная IDS просто отслеживает подозрительную активность, а затем регистрирует ее. В некоторых случаях он может уведомить администратора о произошедшем подозрительном событии. Это самый простой тип IDS. Любая современная система должна иметь, как минимум, пассивную IDS вместе с брандмауэром, антивирусом и другими мерами безопасности.

Активная IDS

Активная IDS или IPS делает дополнительный шаг для закрытия подозрительной связи. Подобно антивирусу, IDS может иметь ложное срабатывание, определяя, что произошла атака, когда на самом деле это законный трафик. Используется ли IDS или IPS, это решение, которое должно быть принято

после тщательного анализа рисков. Например, пусть IPS контролирует превышение объемом трафика некоторого порога, чтобы определить, происходит ли атака. И пусть некоторый пользователь обычно работает между 8:00 и 5:00 и использует относительно небольшой объем полосы пропускания. IPS обнаруживает пользователя в 10 часов вечера, использующего в 10 раз большую полосу пропускания. Тогда IPS отключает подозрительный трафик. Однако позже выясняется, что это был законный пользователь, работавший в конце критического проекта, который был вызван клиентом на следующий день, и данная IPS не позволила ему работать.

Следует взвесить опасность ложных срабатываний против риска того, что атака начнется незамеченной, прежде чем решить, подходит ли для организации пассивная IDS или IPS. Часто бывает, что разные сегменты сети будут иметь разные профили рисков. Можно обнаружить, что пассивная IDS подходит для большей части сети, но для наиболее чувствительных сегментов сети требуется IPS.

4.4.2 Идентификация вторжения

Выделяют *два метода* идентификации вторжения [16, 23, 31, 44]. Первый метод основан на *сигнатурах*, схожих с сигнатурами, используемыми антивирусом. Однако сигнатуры IDS охватывают проблемы, помимо вредоносного ПО. Например, некоторые DoS-атаки имеют определенные сигнатуры, которые можно распознать.

Второй метод – обнаружение *аномалий*. По существу, любая деятельность, которая достаточно сильно отходит от нормальных параметров идентифицируется как вероятная атака. Любое количество действий может инициировать этот тип предупреждения. Примером может быть внезапное увеличение использования полосы пропускания или учетных записей пользователей, получающих доступ к ресурсам, к которым они никогда не обращались. Большинство IDS используют обе формы распознавания атак. Два наиболее важных критерия для выбора IDS – это простота использования и база данных сигнатур.

4.4.3 Элементы IDS

Независимо от того, является ли IDS активной или пассивной, коммерческой или с открытым исходным кодом, некоторые элементы / термины являются общими для всех IDS [31, 38, 43].

– Датчик – это компонент IDS, который собирает данные и передает его анализатору.

– Анализатор – это компонент или процесс, который анализирует данные, собранные датчиком.

– Менеджер – это интерфейс IDS, используемый для управления.

– Оператором является лицо, которое несет основную ответственность за IDS.

– Уведомление – это процесс или метод, с помощью которых менеджер IDS информирует оператора о предупреждении.

– Активность – это элемент источника данных, который представляет интерес для оператора. Активность может быть или не быть возможной атакой.

– Событие – это любая деятельность, которая считается подозрительной и возможной атакой.

– Предупреждение – это сообщение от анализатора, указывающее, что произошло событие.

– Источником данных является необработанная информация, которую IDS анализирует, чтобы определить, произошло ли событие.

Все эти элементы являются частью IDS и работают вместе, чтобы захватить трафик, проанализировать этот трафик и сообщить об аномальной активности оператору IDS. IPS имеет дополнительные элементы, способные отключить нарушающий трафик.

Есть несколько поставщиков систем IDS, каждый из которых имеет свои сильные и слабые стороны. Какая система лучше всего подходит для среды, зависит от многих факторов, включая сетевое окружение, уровень безопасности, бюджетные

ограничения и уровень квалификации человека, который будет работать непосредственно с IDS.

Одной из популярных IDS с открытым исходным кодом является Snort, которую можно бесплатно скачать по адресу www.snort.org.

4.4.4 Обнаружение вторжений веб-приложений

Атаки веб-приложений, такие как инъекции XSS и SQL, можно обнаруживать с помощью инструментов глубокой проверки сетевых пакетов, таких как Bro. Проверка журналов HTTP-сервера может предоставить необходимую информацию для обнаружения вторжений в веб-приложения и является более прямым способом получения характеристик по результатам взаимодействия веб-приложения с пользователем. Стандартные веб-серверы, такие как Apache, IIS и Nginx, генерируют журналы в общем формате файла журнала NCSA (National Center for Supercomputing Applications – Национальный суперкомпьютерный центр), также называемые журналами доступа. Комбинированные журналы NCSA и журналы ошибок также записывают информацию о пользовательском агенте клиента, URL реферала и любых ошибках сервера, генерируемых запросами. В этих журналах каждая строка представляет отдельный HTTP-запрос к серверу, и каждая строка состоит из токенов в четко определенном формате. Вот пример записи в комбинированном формате журнала, который включает в себя пользовательский агент запрашивающей стороны и URL реферала:

123.123.123.123 - jsmith [17 / Dec / 2016: 18: 55: 05 +0800] "GET /index.html HTTP / 1.0" 200 2046 "http://referer.com/" "Mozilla / 5.0 (Macintosh; Intel Mac OS X 10.17.3) AppleWebKit /536.27.14 (KHTML, как Gecko) Chrome / 55.0.2734.24 Safari / 536.27.14 "

В отличие от DPI стандартные журналы веб-доступа не регистрируют данные тела POST. Это означает, что векторы атак, встроенные в пользовательский ввод, не могут быть обнаружены при проверке стандартных журналов доступа.

Самые популярные веб-серверы предоставляют модули и плагины, которые позволяют регистрировать полезные данные HTTP. Модуль Apache mod_dumpio регистрирует все входные и выходные данные, отправленные сервером. Можно добавить директивы proxy_pass или fastcgi_pass в файл конфигурации Nginx, чтобы серверы Nginx заполняли переменную $ request_body фактическим содержимым тела запроса POST.

Microsoft предоставляет серверам IIS расширение Advanced Logging, которое можно настроить для регистрации данных POST.

Даже при сравнительно ограниченном объеме видимости, который предоставляется в стандартных файлах журнала сервера HTTP, есть некоторые интересные характеристики, которые можно извлечь:

– Статистика доступа на уровне IP. Высокая частота, периодичность или объем по одному IP-адресу или подсети является подозрительным.

– Аберрации строки URL. Самостоятельные пути (/./) или обратные ссылки (/../) часто используются при обходе пути. Декодированные сущности URL и HTML, экранированные символы, завершение строки из нулевого байта Они часто используются простыми механизмами подписи / правил, чтобы избежать обнаружения.

– Необычные шаблоны реферера. Доступ к странице с ненормальным URL реферера часто является сигналом нежелательного доступа к конечной точке HTTP

– Последовательность доступа к конечным точкам. Неупорядоченный доступ к конечным точкам HTTP, которые не соответствуют логическому потоку на веб-сайте, указывает на размытое или злонамеренное исследование.

Например, если типичный доступ пользователя к веб-сайту – это POST для / login, за которым следует три последовательных GET для / a, / b и / c, но определенный IP-адрес постоянно делает GET запросы к / b и / c без соответствующего / логина или / запроса, которые могут быть признаком автоматизации ботов или разведывательной деятельности вручную

– Шаблоны пользовательских агентов. Можно выполнять частотный анализ строк пользовательских агентов для оповещения о ранее не замеченных –

увиденных строках пользовательских агентов или крайне старых клиентах (например, пользовательский агент «Mosaic / 0.9» с 1993 года), которые скорее всего подделаны.

Веб-журналы предоставляют достаточно информации для обнаружения различных видов атак на веб-приложения, 10 включая, помимо прочего, первую десятку OWASP - XSS, инъекция, CSRF, небезопасные прямые ссылки на объекты и т. д.

Создание надежного и всеобъемлющего набора признаков имеет решающее значение для процесса обнаружения аномалий. Целью разработки признаков является переработка сложной информации в компактную форму, которая удаляет ненужную информацию, но не жертвует какими-либо важными характеристиками данных. Эти сгенерированные функции будут затем поданы в алгоритмы, которые будут использовать данные и использовать их для обучения моделей.

4.4.5 Honeypot

Honeypot – «методонос», технология, предполагающая, что злоумышленник может нарушить безопасность сети. И было бы лучше отвлечь злоумышленника от ценных данных. Поэтому создается сервер, на котором есть поддельные данные – возможно, сервер SQL или сервер Oracle, загруженный поддельными данными, и немного менее безопасный, чем настоящие серверы [43].

Поскольку ни один из ваших фактических пользователей никогда не обращается к этому серверу, программное обеспечение для мониторинга установлено, чтобы предупредить, когда кто-то обращается к этому серверу. honey pot достигает две цели. Во-первых, отвлекает внимание атакующего от данных, которые нужно защитить. Во-вторых, предоставляет информацию о злоумышленнике, подключившемся к поддельному серверу при его отслеживании.

Существуют коммерческие решения, такие как Spectre (www.specter.com). Эти решения обычно довольно просты в настройке и включают в себя программное обеспечение для мониторинга / отслеживания. Также может быть полезно

ознакомиться с www.honeypots.org для получения дополнительной информации о honeypot в целом и о конкретных реализациях.

4.4.6 Мониторинг активности базы данных

Мониторинг активности базы данных (DAM) – это мониторинг и анализ активности базы данных, работающий независимо от системы управления базами данных (СУБД).

DAM отделен от аудита СУБД, регистрации и мониторинга. Мониторинг активности базы данных и предотвращения (DAMP) является расширением DAM, которое кроме мониторинга и предупреждения, также может блокировать несанкционированные действия [43].

4.4.7 Другие превентивные методы

Помимо IDS, антивирусов, брандмауэров и honey pot существует множество упреждающих методов, которые администратор может использовать, чтобы попытаться уменьшить шансы на успешную атаку, выполняемую против его сети.

Отклонение от вторжения (Intrusion Deflection)

Этот метод становится все более популярным среди администраторов безопасности. Суть его довольно проста. Предпринимается попытка привлечь нарушителя к подсистеме, созданной с целью наблюдения за ним. Это делается путем обмана злоумышленника, полагая, что ему удалось получить доступ к системным ресурсам, когда на самом деле он был направлен в специально разработанную среду. Возможность наблюдать за злоумышленником, когда он практикует свое искусство, даст ценные подсказки и возможность привести к его аресту. Это часто делается с использованием honey pot. По сути, создается фальшивая система, возможно, сервер, который представляется как целая подсеть.

Эту систему делают очень привлекательной, возможно, имеющей конфиденциальные данные, такие как файлы персонала или ценные данные, такие

как номера счетов или исследования. Фактически данные, хранящиеся в этой системе, являются поддельными. Реальная цель системы – внимательно следить за действиями любого человека, который обращается к системе. Поскольку никакой законный пользователь никогда не обращается к этой системе, то любой доступ к ней является нарушающим.

Сдерживание вторжений (Intrusion Deterrence)

Этот метод предполагает просто попытку заставить систему казаться менее приемлемой мишенью. Коротко, делается попытка сделать любую потенциальную награду от успешной попытки вторжения казаться менее значимой, чем она того стоит. Этот подход включает тактику, например, попытку уменьшить кажущуюся ценность текущей системы за счет камуфляжа. Это по существу означает работу по скрытию наиболее ценных аспектов системы. Другая тактика в этой методологии предполагает повышение воспринимаемого риска потенциального вторжения нарушителя. Это можно сделать различными способами, включая заметное отображение предупреждений и предупреждение активного мониторинга. Восприятие безопасности системы может быть значительно улучшено, даже если фактическая безопасность системы не была улучшена.

4.5 Виртуальные частные сети

VPN – это виртуальная частная сеть, представляющая способ использования Интернета для создания виртуального соединения между удаленным пользователем или сайтом и целевым сервером. Пакеты, отправленные туда и обратно по этому соединению, зашифровываются, что делает его закрытым и, следовательно, защищенным. VPN должен эмулировать прямое сетевое соединение [31, 43].

Выделяют следующие протоколы, наиболее широко используемые для создания VPN:

– Протокол туннелирования «точка-точка» (PPTP – Point-to-Point Tunneling Protocol).

– Протокол туннелирования 2-го уровня (L2TP – Layer-2 Tunneling Protocol).

– Безопасность протокола IP (IPsec – Internet Protocol Security).

4.5.1 Протокол туннелирования «точка-точка»

Протокол туннелирования «точка-точка» (PPTP) является самым старым из трех протоколов, используемых в VPN. Он был первоначально разработан как безопасное расширение протокола «точка-точка» (PPP). PPTP был первоначально предложен в качестве стандарта в 1996 году Форумом PPTP – группой компаний, в которую вошли Ascend Communications, ECI Telematics, Microsoft, 3Com и U.S. Robotics. PPTP добавляет функции *шифрования пакетов* и *аутентификации пользователей* к более раннему протоколу PPP. PPTP работает на канальном уровне передачи данных модели OSI.

PPTP предлагает два разных метода аутентификации пользователя: расширяемый протокол аутентификации (EAP – Extensible Authentication Protocol) и протокол аутентификации (CHAP – Challenge Handshake Authentication Protocol) – протокол аутентификации с косвенным согласованием.

EAP был разработан специально для PPTP и не является проприетарным, является фреймворком, предоставляя более 40 различных методов проверки подлинности, используется в том числе в беспроводных сетях. CHAP – это трехсторонний процесс, посредством которого клиент отправляет код на сервер, сервер аутентифицирует его, а затем сервер отвечает клиенту. CHAP также периодически повторяет аутентификацию удаленного клиента после установления соединения, защищает от атак повторного воспроизведения.

PPTP использует шифрование Microsoft Point-to-Point (MPPE) для шифрования пакетов. MPPE – фактически является версией DES. DES по-прежнему полезен для многих ситуаций; однако предпочтительны более новые версии DES, такие как 3DES.

4.5.2 Протокол туннелирования уровня 2

Протокол туннелирования уровня 2 (L2TP) является усовершенствованным PPTP. Как и PPTP, он работает на канальном уровне передачи данных модели OSI. Он имеет несколько улучшений PPTP. Во-первых, он предлагает более разнообразные методы аутентификации – PPTP предлагает два, тогда как L2TP предлагает пять. В дополнение к CHAP и EAP, L2TP предлагает PAP, SPAP и MS-CHAP.

Помимо других протоколов аутентификации, доступных для использования, L2TP предлагает другие улучшения. PPTP может работать только в стандартных IP-сетях, тогда как L2TP может работать в сетях X.25 (общий протокол в телефонных системах) и ATM (асинхронный режим передачи, высокоскоростная сетевая технология). L2TP также использует IPsec для шифрования.

4.5.3 IPsec

IPsec протокол VPN. Одно из отличий между IPsec и двумя другими методами заключается в том, что он шифрует не только пакетные данные, но также и информацию заголовка. Он также имеет защиту от несанкционированной *повторной передачи пакетов*. Это важно, потому что один трюк, который может использовать хакер, – это просто захватить первый пакет из передачи и использовать его, чтобы получить свои собственные передачи для прохождения. По сути, первый пакет (или пакеты) должен содержать данные для входа. Если просто повторно отправить этот пакет (даже если не сможете взломать его шифрование), будут отправлены действующий логин в систему и пароль, за которыми можно посылать любые дополнительные пакеты. Предотвращение несанкционированной повторной передачи пакетов защищает от этого.

IPsec работает в одном из *двух режимов*: *транспортном режиме*, в котором зашифрована только полезная нагрузка, и *режиме туннеля*, в котором шифруются как данные, так и заголовки IP.

Ниже приведены некоторые основные термины IPsec:

– Заголовки аутентификации (Authentication Headers, AH) обеспечивают целостность без установления соединения и аутентификацию источника данных для IP-пакетов.

– Инкапсулирование безопасности (Encapsulating Security Payloads, ESP) обеспечивает подлинность, целостность и конфиденциальность защиты пакетов. Они имеют конфигурации только для шифрования и аутентификации.

– Ассоциации безопасности (Security Associations, SA) предоставляют параметры, необходимые для операций AH или ESP. SA создаются с использованием Ассоциации интернет-безопасности и протокола управления ключами (Internet Security Association and Key Management Protocol, ISAKMP).

– ISAKMP обеспечивают основу для аутентификации и обмена ключами.

– Обмен ключами Интернета (Internet key exchange, IKE и IKEv2) используется для настройки ассоциации безопасности (SA) путем обработки согласований протоколов и алгоритмов и для генерации ключей шифрования и аутентификации, которые будут использоваться.

По существу, во время первоначального создания туннеля IPsec формируются ассоциации безопасности (SA). Эти SA содержат информацию, такую как алгоритм шифрования и алгоритмы хеширования, которые будут использоваться в туннеле IPsec. IKE в первую очередь занимается установкой этих SA. ISAKMP позволяет двум концам туннеля IPsec аутентифицироваться друг друга и обмениваться ключами.

4.6 Безопасность Wi-Fi

Поскольку беспроводные сети настолько распространены, важно учитывать безопасность беспроводной сети. Существует три протокола специально разработанных для безопасности Wi-Fi, начиная от самого старого и наименее безопасного (WEP) до самого последнего и самого безопасного (WPA2). Они кратко описаны далее [43].

WEP

(Wired Equivalent Privacy, WEP) использует шифр потока RC4 для защиты данных и контрольной суммы CRC-32 для проверки ошибок. Стандартный WEP использует 40-битный ключ (известный как WEP-40) с 24-битным вектором инициализации (IV) для эффективного формирования 64-битного шифрования. 128-битный WEP использует 104-битный ключ с 24-битным IV. Поскольку RC4 – это потоковый шифр, один и тот же самый ключ трафика никогда не должен использоваться дважды. Проблема с WEP заключается в том, что комитет, создавший его, состоял из очень хороших компьютерных специалистов, которые думали, что они достаточно разбираются в криптографии, но этого не сделали и повторно используют IV, что нивелирует всю цель IV и оставляет протокол открытым для атак. Простой запрос в YouTube вида «как взломать WEP» приведет к потоку видео по технике взлома WEP.

WPA

Wi-Fi Protected Access (WPA) определенно был улучшен по сравнению с WEP. Во-первых, WPA использует AES, являющимся криптостойким алгоритмом шифрования. Затем WPA использует протокол Temporal Key Integrity Protocol (TKIP). TKIP динамически генерирует новый ключ для каждого пакета. Таким образом, даже если взломают ключ WPA, будет сгенерирован новый ключ для следующего пакета.

WPA2

Это самая современная форма безопасности Wi-Fi, и если это вообще возможно, это то, что следует использовать.

WPA2 основан на стандарте IEEE 802.11i. Он обеспечивает расширенный стандарт шифрования (AES) с использованием режима CBC – протокола шифрования – Message Authentication Code MAC (CCMP), который обеспечивает конфиденциальность данных, аутентификацию источника данных и целостность данных для беспроводных кадров. CBC предотвращает известные атаки обычного текста. MAC сохраняет целостность сообщений и гарантирует, что пакеты не были

изменены при транзите, случайно или намеренно. Это означает, что WPA2 использует очень сильное шифрование наряду с целостностью сообщений.

Таким образом, подводя итоги, следует отметить, что крайне важно, что в любой сети есть межсетевой экран и прокси-сервер между сетью и внешним миром. Очень важно, чтобы все компьютеры в сети (серверы и рабочие станции) обновляли защиту от вирусов. Также рекомендуется рассмотреть необходимость применения IDS и антишпионских программ.

Контрольные вопросы

1 Какими способами работают антивирусы?

2 В чем заключается анализ вредоносных программ?

3 Что такое обратный инжиниринг?

4 Как можно классифицировать вредоносное программное обеспечение?

5 Для чего предназначены межсетевые экраны?

6 Какие существуют типы межсетевых экранов?

7 Как работают инспекторы состояния пакетов?

8 Перечислите основные виды конфигураций / реализаций межсетевых экранов.

9 Понятие IDS? Для чего предназначены IDS?

10 Назовите основные компоненты IDS.

11 Особенности обнаружения вторжений веб-приложений.

12 В чем заключается технология Honeypot?

13 Какие превентивные методы можно использовать, чтобы уменьшить шансы успешной атаки?

14 Объясните суть технологии VPN.

15 Каким образом применяется в VPN протокол L2TP?

16 В чем заключаются отличия протокола IPsec от других протоколов VPN?

17 Какие протоколы специально разработаны для безопасности Wi-Fi?

5 Безопасность распределенных приложений

5.1 Безопасность баз данных

Базы данных (БД) часто являются важными элементами КС и программных приложений. Поскольку БД играют важную роль в хранении большого количества потенциально ценной информации, они часто становятся объектом атак со стороны злоумышленников, стремящихся получить доступ к этим данным [22, 39, 48].

Кроме того, БД часто содержат конфиденциальную информацию, которая может также раскрывать информацию о людях, поэтому другой проблемой безопасности в отношении БД является конфиденциальность.

5.1.1 Двухфазная фиксация

Одной из серьезных проблем, позволяющих нескольким агентам одновременно обновлять БД в распределенной среде в сети, является то, что *операции обновления* могут *конфликтовать*. Например, возникает проблема, когда пользователь-1 хочет удалить запись, и пользователь-2 хочет изменить одно из значений атрибута той же самой записи одновременно. Кроме того, даже если несколько одновременных обновлений не конфликтуют, есть вероятность, что во время одного из этих обновлений может произойти сбой компьютера или сети, и обновление завершится не полностью. Такой отказ может оставить БД в *несогласованном состоянии*, что может даже сделать ее непригодной. Для решения проблем согласованности и надежности, большинство БД используют протокол, называемый *двухфазной фиксацией* для выполнения обновлений.

Последовательность операций протекает по двум этапам:

1 Первая фаза – это *фаза запроса*, в которой все части БД, которые необходимо изменить в результате этого обновления, идентифицируются и помечаются как предназначенные для этого изменения. Результатом этого этапа является либо то, что он завершается успешно, и все запрошенные изменения доступны и теперь помечены для изменения, или этап прерывается, так как

невозможно отметить все нужные для изменения части (например, потому что кто-то уже отметил их) или из-за сбоя сети или системы. Если первый этап прерывается, то все его запрошенные изменения сбрасываются, что всегда возможно, потому что никаких постоянных изменений еще не было сделано. Если первая фаза завершается успешно, протокол продолжается во второй фазе.

2 Второй фазой является *фаза фиксации*, в которой БД блокируется в режиме изменения и выполняется последовательность изменений, которые были идентифицированы на этапе запроса. Если фаза завершается успешно, очищаются все флаги, идентифицирующие запрошенные изменения, и блокировка БД снимается. Если эта операция завершается с ошибкой, тогда она *откатывается*, то есть все внесенные изменения, изменяются обратно в состояние, в котором находилась БД, после завершения первой фазы.

Таким образом, протокол двухфазной фиксации является функцией, которую БД может использовать для обеспечения *целостности и доступности*. Он поддерживает целостность, поскольку БД всегда либо находится в согласованном состоянии, либо может быть отброшена в согласованное состояние. Этот протокол также поддерживает доступность, поскольку БД никогда не попадает в состояние внутренней несогласованности, которая может привести к сбою системы управления БД.

5.1.2 Контроль доступа к базе данных

БД используют несколько мер безопасности для предотвращения атак, защиты конфиденциальной информации и создания модели безопасности, которая минимизирует возможность компрометации БД. Хотя детали реализации зависят от БД, большинство БД предоставляют систему контроля доступа, которая позволяет администраторам точно определять, что некоторым пользователям и группам разрешено делать в отношении этой БД. Например, многие системы реализуют схему *списка управления доступом* (ACL), аналогичную схеме, используемой операционными системами. Простая система контроля доступа может позволить,

например, веб-приложению выполнять поисковые запросы по данным и вставлять новые записи, но не создавать или удалять таблицы или выполнять системные команды через БД. Более сложные наборы правил также могут использоваться для определения разных наборов разрешений для нескольких пользователей. Например, БД, которая включает в себя таблицы студенческих записей и записей о занятости в университете, может позволить преподавателям добавлять и обновлять оценки для учащихся, но не позволять им вносить изменения в свои собственные записи о занятости. С другой стороны, декану могут быть предоставлены права на внесение дополнений и модификаций как для записей учащихся, так и для сотрудников. В целом, возможность определять разрешения доступа для различных пользователей БД помогает минимизировать ущерб от инсайдерских атак, таких как утечка информации посредством чрезмерно любопытных сотрудников или учащихся, которые пытаются изменить оценки в своих транскриптах. Правильно определенные разрешения доступа также могут быть критически важной *превентивной мерой* компрометации БД в случае вторжения. Например, рассмотрите БД, в которой хранится информация для двух разделов веб-сайта новостей, статей и фотографий на основе подписки в одном разделе и финансовых отчетов о клиентах в другом разделе (например, номера кредитных карт). В этом случае БД и веб-приложение должны быть настроены так, чтобы каждая часть приложения имела доступ только к необходимой информации для этой части. С учетом этой меры безопасности, если непривилегированный раздел новостей на веб-сайте будет скомпрометирован, злоумышленник не сможет получить доступ к конфиденциальной информации о клиентах.

Таким образом, создавая права доступа с использованием понятий наименьших привилегий и разделения привилегий, ущерб от вторжений может быть сведен к минимуму. Собственный набор элементов управления доступом должен реализовывать принцип наименьших привилегий, чтобы каждый пользователь имел необходимые права для выполнения своих требуемых задач, но не имел никаких других прав.

5.1.3 Контроль доступа

Использование SQL. SQL определяет структуру управления доступом, которая обычно используется для определения привилегий БД. Когда таблица создается, владелец таблицы имеет единственные права на выполнение операций над этой таблицей. Затем владелец может предоставить привилегии другим пользователям, которые известны как делегирование полномочий. Эти привилегии могут быть широкими, например, возможность делать что-либо в определенной таблице или мелкозернистую операцию, например, возможность выполнять только запросы SELECT в определенных столбцах. Например, владелец таблицы может выдавать следующую команду SQL, чтобы дать Вере возможность искать сотрудников таблицы:

GRANT SELECT ON employees To Vera;

Другие разрешения, которые могут быть предоставлены с использованием ключевого слова GRANT, включают DELETE, INSERT и UPDATE. Кроме того, для предоставления всех доступных прав можно использовать ключевое слово ALL. Разрешения могут предоставляться отдельным лицам или всем (с использованием ключевого слова PUBLIC). Кроме того, разрешения могут предоставляться для *ролей*, что позволяет использовать управление доступом на основе ролей для БД. Кроме того, владелец таблицы может создать виртуальное подмножество данных, известных как представление, к которым затем могут обращаться другие пользователи. Например, владелец таблицы может пожелать разрешить пользователю, Вере, обновлять только свою собственную информацию. Это можно сделать, создав представление об общем наборе данных, который включает только данные Веры, и предоставление доступа для обновления в этом представлении Веры.

Привилегированное делегирование и аннулирование.

Помимо возможности предоставления определенных привилегий другим пользователям, владельцы таблиц также могут разрешать другим пользователям предоставлять привилегии для этих таблиц, которые известны как политика

делегирования полномочий. В частности, при предоставлении привилегии пользователю, как в приведенных выше примерах, лицо, предоставляющее право, может включать предложение WITH GRANT OPTION, чтобы предоставить получателю возможность дополнительно делегировать эту привилегию. Например, администратор может создать представление для Веры и дать ей разрешение делегировать разрешения SELECT для этого представления другим пользователям следующим образом:

CREATE VIEW employees_vera AS

Правильное выполнение делегирования и аннулирования прав требует дополнительных затрат. Формальное значение *отзыва привилегий* заключается в том, что привилегии, предоставляемые пользователям, должны быть такими же, как если бы отмененная привилегия никогда не предоставлялась. Перевыполнение всех привилегий для каждого пользователя с нуля путем повторного воспроизведения всех заявлений GRANT, когда-либо выпущенных, за исключением отозванного, является вычислительно очень обременительным.

5.1.4 Чувствительные данные

В дополнение к обеспечению того, чтобы БД имели соответствующие меры контроля доступа, необходимо проявлять осторожность, чтобы гарантировать хранение конфиденциальных данных таким образом, чтобы защитить конфиденциальность пользователей и любые требования конфиденциальности для данных.

Использование криптографии

Если информация, хранящаяся в БД, имеет требования к конфиденциальности, она не должна храниться в открытом тексте, а должна храниться как результат криптографической функции [21].

Защита конфиденциальности

Помимо мер, направленных на защиту конфиденциальности информации пользователя, владельцы БД должны рассмотреть влияние публикации на конфиденциальность или предоставление доступа к конфиденциальной информации. Если БД должна публиковаться открыто, например, для использования в исследовательских целях, то вся идентифицирующая информация, такая как имена, адреса, номера социального страхования, номера сотрудников и номера учащихся, должна быть удалена или изменена на маскирующие значения, которые являются неопределенными значениями, которые не содержат всей идентифицирующей информации. Например, БД сотрудников может быть обнародована после замены каждого имени сотрудника уникальным идентификатором, таким как id001, id002, id003 и т.д.

Хранение паролей для учетных записей пользователей в БД. Эти пароли никогда не должны храниться в открытом виде, так как вторжение может привести к компрометации каждой учетной записи пользователя. Вместо этого следует хранить криптографический хэш каждого пароля и идентификатор. Когда пользователь пытается войти в систему, пароль, предоставленный пользователем, и идентификатор, хранящаяся в БД, будут хешированы и сопоставлены с сохраненным значением хэш-функции. Таким образом, если злоумышленник скомпрометировал бы БД, он приобрел бы список хэшей, из которых невозможно было восстановить фактические пароли, если бы не была успешной атака словаря или грубой силы.

Атаки логического вывода

Даже если идентификационная информация удалена или замаскирована, все равно можно использовать БД вместе с дополнительной информацией, доступной злоумышленнику, чтобы узнать больше о данных базы. Это *называется атакой логического вывода*. В качестве примера рассмотрим БД записей сотрудников, атрибутами которых являются имя, пол, идентификационный номер и зарплата. Предположим, что стороне предоставлен доступ к санитарной версии таблицы, где атрибут имени удален, с целью создания статистики о заработной плате по полу. У другой стороны может быть список пар, связывающих идентификационные номера

с именами для задачи отчетности. Если бы эти две стороны должны были общаться, они могли бы легко определить заработную плату каждого сотрудника, несмотря на намерение владельца БД. В общем, при предоставлении доступа к модифицированным версиям базы данных администраторы должны учитывать, может ли сговор между грантополучателями позволить им получить несанкционированную информацию.

Защита баз данных от атак с помощью меток

Чтобы защитить БД от атак логического вывода, могут быть использованы следующие методы до публикации БД.

– Подавление ячеек. При использовании этого метода некоторые ячейки в БД удаляются и остаются пустыми в опубликованной версии. Цель состоит в том, чтобы подавить критические ячейки, которые можно было бы использовать в атаках логического вывода, чтобы определить чувствительные данные для людей (см. рисунок 5.1б).

– Обобщение. При использовании этого метода некоторые значения в опубликованной БД заменяются более общими значениями. Например, дата рождения, например, «2 июня 1983 года», может быть заменена на некоторый диапазон лет, например, «1980-1984», почтовый индекс, например, «92697-3435», может быть изменен на «926xx-xxxx». Цель состоит в том, чтобы обобщить критические значения, для перемешивания с другими значениями, чтобы сделать атаки логического вывода менее осуществимыми (см. рисунок 5.1 в).

– Добавление шума. При использовании этого метода к значениям в опубликованной БД добавляют случайные значения, так что шум по всем записям для одного и того же атрибута в среднем равен нулю. Например, к возрасту можно прибавить случайное значение в диапазоне от -5 до 5. Цель состоит в том, чтобы затенять отдельные значения, оставив среднее значение неизменным. Конечно, все эти методы делают информацию в опубликованной БД менее конкретной (см. рисунок 5.1г).

Num	Age1	Age2
11	49.3	53.6
18	46.9	63.2
20	49.3	49.8
35	43.6	46.5
42	46.4	
44	47.5	

а)

Num	Age1	Age2
11	49.3	
18	46.9	63.2
20	49.3	
35		
42	46.4	
44	47.5	

б)

Num	Age1	Age2
11	45–50	50–60
18	45–50	60–75
20	45–50	45–50
35	40–45	45–50
42	45–50	
44	45–50	

в)

Num	Age1	Age2
11	47.7	55.2
18	49.2	64.3
20	51.6	52.8
35	42.3	47.3
42	47.1	
44	48.0	

г)

Рисунок 5.1 – Методы защиты

Учитывая описанные выше методы затемнения данных, возникает вопрос о том, как далеко можно продвинуться в их применении, обеспечат ли они достаточную защиту конфиденциальности. В крайнем случае, можно «размыть» данные настолько, чтобы сделать их совершенно бесполезными, с немного большим смыслом, чем БД случайного шума и пустых ячеек. Это защитит конфиденциальность данных, но также сделает их совершенно бесполезными. Таким образом, следует применять методы затемнения в сочетании с некоторым правилом для принятия решения о том, что данные были достаточно затенены. Однако, до сих пор нет общепринятого стандарта для принятия решения о том, что информация в публичной БД недостаточно затенена.

Тем не менее, предлагаются следующие правила:

– *k-анонимизация*. В этом стандарте БД считается достаточно анонимной, если любой возможный запрос SELECT возвращает не менее *k* записей, где **k** является достаточно большим порогом допускаемости раскрытия.

– *Дифференциальная конфиденциальность*. В этом стандарте БД считается достаточно анонимной, если для любой записи R в БД вероятность p для некоторого чувствительного свойства P, в случае если R *находится* в БД, и

вероятность p_0 для свойства P, причем R *не находится* в БД, отличаются не более чем на e , где e - достаточно маленький порог допуска утечки информации.

Оба этих свойства обеспечивают количественную оценку уровня конфиденциальности.

5.2 Безопасность электронной почты

Электронная почта является одним из наиболее широко используемых интернет-приложений. Из-за этого широкого и повсеместного использования электронной почты для её защиты требуется, обсудить несколько классических проблем безопасности, включая аутентификацию, целостность и конфиденциальность, а также важную проблему безопасности, связанную с спамом электронной почты.

5.2.1 Протоколы работы электронной почты

Системы электронной почты используют несколько протоколов для доставки сообщений. Чтобы обрабатывать отправку сообщений с компьютера клиента на почтовый сервер получателя, используется протокол Simple Mail Transfer Protocol (SMTP). SMTP – это простой текстовый протокол уровня приложения, который использует TCP для облегчения взаимодействия между клиентом, желающим отправить почту, и соответствующим сервером-получателем. В модели SMTP клиент называется *агентом почтового пользователя* (Mail User Agent, MUA). MUA отправляет SMTP-сообщение *агенту отправки почты* (Mail Sending Agent, MSA), который, в свою очередь, доставляет сообщение *агенту передачи почты* (Mail Transfer Agent, MTA), ответственному за передачу сообщения принимающей стороне. MSA и MTA часто находятся на одном физическом сервере. Сообщение передается из MTA отправителя в MTA получателя, где он передается *агенту доставки почты* (Mail Delivery Agent, MDA), ответственному за обеспечение того, чтобы сообщение доходило до MUA получателя [2, 11].

Сеанс Client-Server

Клиент инициирует сеанс SMTP через порт 25 с MSA, например, управляемым интернет-провайдером пользователя. После установления соединения TCP и получения заголовка сервера клиент идентифицирует себя с помощью команды HELO. После получения подтверждения от сервера клиент идентифицирует отправителя сообщения с полем MAIL FROM.

Затем клиент указывает получателей, используя поле RCPT TO. Наконец, клиент предоставляет сообщение и любые вложения в разделе DATA, после чего сообщение отправляется, и клиент завершает соединение с командой QUIT. Пример SMTP-переговора может выглядеть следующим образом, где клиент обозначается как «C», а сервер - «S»:

```
S: 220 mail.example.com ESMTP Postfix
C: HELO relay.example.com
S: 250 mail.example.com Hello relay.example.com, pleased to meet you
C: MAIL FROM:<joe@example.com>
S: 250 <joe@example.com> sender ok
C: RCPT TO:<alice@othersite.com>
S: 250 <alice@othersite.com> recipient ok
C: DATA
S: 354 enter mail, end with "." on a line by itself
C: From: "Joe Smith" <joe@example.com>
C: To: "Alice" <alice@othersite.com>
C: Subject: Sample SMTP conversation
C: This is an example of an SMTP conversation. Hope you like it.
C: .
S: 250 Mail accepted for delivery
C: QUIT
S: 221 mail.example.com closing connection
```

Затем MSA отправляет это сообщение в MTA, который затем запрашивает у системы доменных имен (DNS) разрешение IP-адреса MTA получателя.

Например, joe@example.com данный получатель, MTA отправителя получит IP-адрес для MTA от домена example.com. Затем MTA отправителя отправляет сообщение получателю MTA с аналогичным переговором, как указано выше, и MTA передает сообщение в MDA.

Протокол SMTP обрабатывает отправку почты на серверы, предназначенные для обработки очередей сообщений, но не используется для доставки почты клиентам. Вместо этого в основном используются два других протокола: *протокол почтового отделения* (Post Office Protocol, POP) и *протокол доступа к интернет-*

сообщениям (Internet Message Access Protocol, IMAP). POP старее из этих двух и был разработан для поддержки клиентов с подключением к Интернету через dial-up. Таким образом, типичный диалог POP включает в себя подключение клиента к их MDA, загрузку любых новых сообщений, удаление этих сообщений с сервера и отключение. IMAP – это более новый протокол, который обеспечивает как онлайн, так и автономную работу. В онлайн-режиме клиент подключается к почтовому серверу и поддерживает постоянное соединение, которое позволяет ему загружать сообщения по мере необходимости. IMAP также позволяет клиентам выполнять поиск сообщений на почтовом сервере по нескольким критериям до фактической загрузки этих сообщений. Наконец, большинство сеансов IMAP по умолчанию оставляют все почтовые сообщения на почтовом сервере, а не удаляют их при загрузке.

5.2.2 Шифрование и аутентификация

Ни один из вышеперечисленных протоколов для отправки и получения электронной почты не имеет встроенного механизма, гарантирующего конфиденциальность сообщений электронной почты. Следовательно, любая сторона, способная перехватывать трафик посредством IP-сниффинга, сможет прослушивать любые передаваемые сообщения электронной почты в своей подсети. Для обеспечения конфиденциальности электронная почта может быть *зашифрована* одним из двух способов: на транспортном уровне или на прикладном уровне.

Самым распространенным методом защиты конфиденциальности электронной почты является шифрование фактической передачи сообщений, а не их содержимого. Большинство почтовых серверов поддерживают использование SSL / TLS, протоколов, которые надежно шифруют TCP-трафик. Эти протоколы часто используются на каждом уровне связи – между клиентом и локальным почтовым сервером, между локальным и целевым почтовыми серверами, а также между целевым почтовым сервером и получателем.

Полагаясь исключительно на шифрование транспортного уровня, защищает сообщения от перехвата в полете, но подразумевает уровень доверия к почтовым серверам, обрабатывающим эти сообщения. Например, сотрудник интернет-провайдера, имеющий доступ к почтовому серверу этого интернет-провайдера, может прочитать содержимое всех сообщений электронной почты, хранящихся на этом сервере.

PGP

Чтобы обеспечить более высокий уровень конфиденциальности, который защищает сообщения от клиента к клиенту, должно быть зашифровано фактическое содержимое сообщения электронной почты. Существует несколько подходов, которые были предложены для этой цели. Одна известная система - Pretty Good Privacy (PGP), которая использует криптосистему открытых ключей для шифрования и / или цифровой подписи сообщений электронной почты. При отправке сообщения предполагаемому получателю с использованием PGP отправитель шифрует сообщение с использованием открытого ключа получателя, так что только получатель может расшифровать сообщение, используя его соответствующий закрытый ключ. Проверка подлинности открытого ключа получателя важна для безопасности PGP, так как в противном случае злоумышленник может потенциально обмануть отправителя в использовании открытого ключа злоумышленника, для которого у него есть соответствующий закрытый ключ. PGP опирается на понятие сети доверия, в отличие от иерархической модели, используемой службами сертификации, такими как SSL. Вместо того, чтобы использовать цепочку, ведущую к доверенному корневому сертификату, PGP использует схему, в которой каждый открытый ключ может быть цифровой подписью других доверенных пользователей, известных как интродьюсеры, чтобы подтвердить, что открытый ключ фактически принадлежит стороне, требующей права собственности. Основная идея заключается в том, что после использования системы в течение длительного периода времени каждый пользователь сохранит коллекцию доверенных ключей, и каждая соответствующая

доверенная сторона может взять на себя роль интродьюсера и проверить подлинность нового открытого ключа.

Аутентификация

В настоящее время для аутентификации источника сообщения электронной почты используются два основных подхода:

– *Аутентификация пользователя-отправителя*. Этот подход позволяет почтовому серверу получателя идентифицировать автора сообщения электронной почты. Однако для его эффективности требуется широкое распространение пар частных и открытых ключей для почтовых пользователей. По этой причине его редко используют на практике.

– *Аутентификация отправляющего агента передачи почты*. В том подходе обычно идентифицируется организация автора, но не отдельный автор. Его проще разворачивать, чем аутентификацию пользователя-отправителя и развивать. Осложнение может возникнуть со всеми типами подписанных сообщений электронной почты, поскольку даже незначительные модификации во время транзита, такие как изменение кодировки, приведут к сбою проверки подписи. Таким образом, тело подписанных сообщений электронной почты должно быть отформатировано таким образом, чтобы уменьшить риск изменений во время транспортировки. Этот процесс форматирования называется *канонизацией*.

Аутентификации пользователя-отправителя: S / MIME.

Сообщение электронной почты может быть подписано цифровой подписью для аутентификации отправителя. Для этого подхода к работе MUA отправителя и получателя должны поддерживаться криптографические операции, связанные с подписью и проверкой, и должна согласоваться используемая криптосистема. Проверка подписанного сообщения электронной почты основывается на знании получателем открытого ключа отправителя. Этот ключ может быть доставлен получателю через безопасный канал или может быть подтвержден доверенным лицом получателя. В стандарте S / MIME для аутентификации пользователя-отправителя электронное письмо структурировано в соответствии со стандартом

многоцелевых расширений почты Интернета MIME (Multipurpose Internet Mail Extensions), который определяет формат и кодировку вложений [2].

Сообщение S / MIME имеет тело, состоящее из двух частей:

– Первая часть – это само сообщение, которое может состоять в свою очередь из нескольких частей, таких как текст и вложения.

– Вторая часть – это подпись над первой частью. Структура сообщения S / MIME показана на схематическом примере на рисунке 5.2 (a).

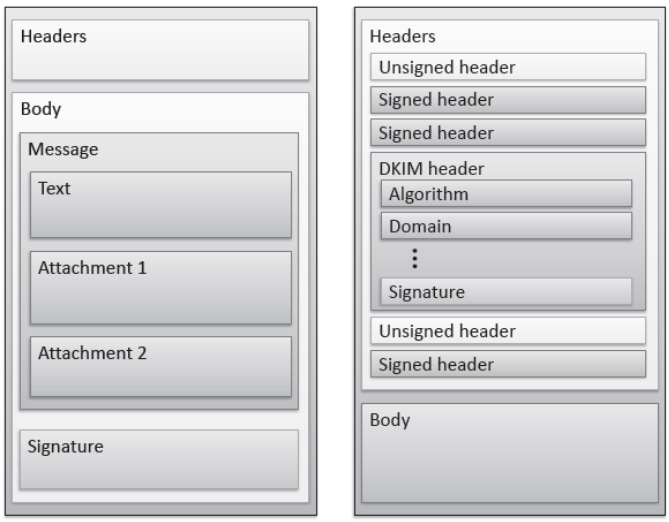


Рисунок 5.2 – Сообщения электронной почты с цифровой подписью: (a) Структура сообщения S / MIME, где часть подписи относится к остальной части тела сообщения, но не к заголовкам. (b) Структура сообщения DKIM, где подпись в поле заголовка DKIM относится к телу сообщения и выбранным заголовкам.

Аутентификации агента отправителя MTA: DKIM.

Первым подходом для аутентификации отправляющего агента передачи почты (MTA) является DomainKeys Identified Mail (DKIM). В DKIM подписывающий объект, обычно MTA отправителя, добавляет подпись к сообщению, чтобы указать, что он произошел из домена подписавшего объекта. DKIM полагается на DNS. Подпись DKIM охватывает не только тело сообщения, но и выбранные заголовки. В частности, поле FROM должно быть подписано. Подпись включена в специальное поле заголовка под названием DKIM Signature, которое добавляется к сообщению. Атрибуты DKIM-подписи включают в себя следующее:

– v: версия спецификации DKIM;

– d: домен объекта подписи;

– s: селектор ключа подписи в домене;

– a: идентификатор криптографических алгоритмов, используемых для подписи и хэширования, например, rsa-sha256;

– c: алгоритм канонизации, преобразование, применяемое к сообщению, для стандартизации его формата (например, удаление пустых строк в конце) перед хэшированием;

– h: список полей заголовка, охватываемых сигнатурой, в дополнение к body;

– bh: hash тела сообщения;

– b: подпись.

Таким образом, DKIM уязвим для атак на инфраструктуру DNS, если DNSSEC не развернут.

Преимущества аутентификации MTA

Одним из небезопасных аспектов электронной почты, связанным с ее созданием, является то, что без проверки подлинности MTA поле FROM в сообщении электронной почты может быть настроено на все, что нравится отправителю. Таким образом, если отправитель утверждает, что является надежным финансовым учреждением, в стандартном протоколе нет ничего, для предотвращения этого. Преимущество аутентификации MTA заключается в том, что она затрудняет отправку фальсифицированного поля FROM, поскольку MTA должен быть готов подписать это поле как действительное для отправителей, за которые отвечает MTA. Все чаще службы электронной почты, такие как Gmail, применяют DKIM, чтобы подписывать как теги, так и заголовки сообщений исходящего контента. Кроме того, многие службы электронной почты начали отклонять сообщения, которые не были подписаны цифровой подписью. Например, Gmail теперь отклоняет все сообщения, заявляющие, что они принадлежат доменам eBay и PayPal, если у них нет действительной подписи DKIM, подтверждающей их происхождение. Эти шаги эффективны для устранения аутентификации MTA отправителя: SPF и SIDF. Политика защиты отправителя (Sender Policy Framework – SPF) следует альтернативному подходу к аутентификации отправляющего MTA, где

криптография не используется. IP-адреса MTA, уполномоченных отправлять почту для домена, сохраняются в текстовой записи DNS для этого домена. Принимающий MTA проверяет, что IP передающей MTA находится в списке разрешенных IP-адресов для домена отправителя, как указано в команде MAIL FROM SMTP. SPF полагается на IP-адрес отправляющего MTA. Таким образом, он уязвим для IP-спуфинга атак и DNS-атаки отравления кэша. Ограничение SPF заключается в том, что оно не поддерживает пересылку почты. Кроме того, SPF не защищает целостность тела сообщения. Преимущества SPF над DKIM заключаются в более быстрой обработке и более простом выполнении из-за отсутствия криптографических операций в отправляющих и принимающих MTA. Недостатки SPF над DKIM включают отсутствие поддержки для пересылки почты и целостности контента. И SPF, и DKIM уязвимы для атак на инфраструктуру DNS.

Структура идентификатора отправителя (Sender ID Framework – SIDF) похожа на SPF. Он также проверяет домен отправителя, указанный в заголовке, например, в полях FROM или SENDER.

5.2.3 Спам

С самых ранних дней электронной почты рекламодатели пытались извлечь выгоду из легкости, с которой электронная почта позволяет получить доступ к миллионам потенциальных клиентов. Электронная почта спама, формально называемая *незапрашиваемой массовой электронной почтой*, представляет собой любую форму электронной почты, которая отправляется многим получателям без предварительного контакта. Спам чаще всего содержит рекламные объявления, но также может иметь более скверные мотивы, такие как фишинг и другие попытки совершить мошенничество. В зависимости от страны спам может быть сомнительно законным, но применение законов, запрещающих отправку спама, оказалось сложным, учитывая глобальный характер проблемы. Спам настолько распространен, что на него, по оценкам, приходится около 94% всех отправленных сообщений. Для рекламодателей спам привлекателен тем, что в отличие от неэлектронной почты

большая часть расходов, связанных с отправкой спама, размещается у получателей, которые вынуждены хранить и обрабатывать электронную почту. Для крупных организаций эта стоимость не является тривиальной, по оценкам, спам стоит предприятиям около 100 миллиардов долларов в год. Помимо этого огромного финансового бремени спам может стать проблемой для конечного пользователя, от неудобства до прямой угрозы. Спам часто является вектором атак для мошенников, средством распространения вредоносного ПО по электронной почте, отправной точкой для фишинговых атак или попыткой социальной инженерии в надежде привлечь получателя к выполнению некоторых недобросовестных действий. Из-за этих факторов был разработан широкий спектр технологий для борьбы со спамом и предотвращения его доступа к конечному пользователю. Далее рассматриваются некоторые методы, используемые спамерами, и некоторые профилактические меры, которые могут быть применены для борьбы со спамом.

Адреса для рассылки

Существует несколько методов, с помощью которых спамеры получают списки рассылки. Некоторые автоматически собирают адреса, используя специально разработанные программы, которые сканируют Интернет и собирают все, что похоже на адрес электронной почты, процесс, известный как робот-паук (*spidering).* Отдельные лица могут часто препятствовать несложным спам-комбайнам, просто отправляя свой адрес электронной почты в измененной форме, например, например, john (dot) smith (at) example (dot) com, который легко понимается людьми, но может быть трудно обнаружаем автоматически. В дополнение к автоматическому поиску адресов электронной почты спамеры часто покупают и продают списки адресов электронной почты от других спамеров, рекламных партнеров или криминальных сетей. По этой причине пользователям рекомендуется указывать адрес электронной почты только доверенным лицам и просматривать политику конфиденциальности любого веб-сайта при принятии решения о том, следует ли указывать адрес электронной почты на этом веб-сайте.

Отправка спама. Спамеры используют множество методов, облегчающих отправку огромного количества электронной почты. Наиболее распространенный

метод заключается в скрытии происхождения электронной почты путем *простого подмены поля FROM* сообщения. Хотя это может обмануть среднего получателя, IP-адрес SMTP-сервера отправителя также включен в заголовок электронной почты, поэтому любое дальнейшее расследование выявит это подмену.

Открытые реле и прокси-серверы. Если спамеры отправили почту с почтового сервера ISP напрямую, получатели, скорее всего, будут жаловаться на этого провайдера, который, в свою очередь, отключит учетные записи спамера. Вместо этого большинство спамеров добавляют слой ложного направления, отправляя спам через третью сторону. Открытое почтовое реле – это SMTP-сервер, который настроен на отправку электронной почты от любого получателя в любой пункт назначения, в отличие от большинства почтовых серверов ISP, которые отправляют только электронную почту от имени своих клиентов. Спамеры могут использовать открытые реле для отправки своей почты, не полагаясь на почтовые серверы ISP. Тем не менее, опасность запуска открытого реле широко признана, поэтому сегодня очень мало почтовых серверов допускают такое поведение. Другой распространенный метод, используемый спамерами, основан *на прокси-серверах*, то есть на серверах, которые выступают посредниками в выполнении соединений между парами пользователей Интернета. Например, когда одна сторона отправляет другой стороне сообщение через прокси-сервер, сообщение подается получателю из прокси, а не из истинного источника. Открытые прокси – это серверы с такой функциональностью могут свободно использоваться кем угодно в Интернете. Отправляя почту через открытые прокси-серверы, спамеры могут скрывать истинный источник своих сообщений. Чтобы отследить спам до его источника, следователям необходимо будет анализировать журналы с прокси-серверов, которые могут находиться в любой точке мира и не могут сотрудничать без вмешательства правительства. В то время как открытые почтовые ретрансляторы служат нескольким законным целям, открытые прокси обычно размещаются людьми, желающими предоставить пользователям возможность анонимно просматривать Интернет, что по своей сути небезопасно или злонамеренно.

CAPTCHAs Растущая популярность веб-почты предоставила спамерам новую стратегию. Спамеры могут просто зарегистрировать учетную запись в бесплатной службе веб-почты и использовать эту учетную запись для отправки спама, пока поставщик веб-почты не обнаружит это действие. Многие спамеры автоматизировали этот процесс, создав программы, которые регистрируют учетные записи электронной почты, отправляют как можно больше почты и повторяют процесс, когда учетная запись отменяется.

Для борьбы с автоматизированной тактикой создания электронной почты большинство служб электронной почты требуют, чтобы пользователи решали CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart – полностью автоматизированный публичный тест Тьюринга для различения компьютеров и людей). Такая задача – это то, что легко разрешается человеком, но сложно решить программным путем. Большинство CAPTCHA – проблемы распознавания образов, где представлен искаженный образ, содержащий строку текста, и пользователь должен интерпретировать встроенный текст.

Просить пользователя ввести слова, которые они видят внутри прямоугольников, в указанном порядке, это то, что относительно легко сделать для человека по сравнению с компьютером. К сожалению, некоторые спамеры обходят эти CAPTCHA с помощью веб-сайтов, которые требуют, чтобы посетители решали CAPTCHA, чтобы получить доступ. Без ведома посетителей эти CAPTCHA фактически копируются с страниц регистрации веб-почты. Затем пользовательские решения передаются на автоматические спам-боты, чтобы зарегистрировать учетную запись электронной почты для отправки спама. Кроме того, некоторые спамеры даже используют низкооплачиваемых рабочих из развивающихся стран для решения CAPTCHA для них. Однако в любом случае использование CAPTCHA увеличивает эксплуатационные расходы спамеров; следовательно, эти методы имеют положительный эффект.

Спам и вредоносное ПО

Часто компьютеры, зараженные вредоносными программами, используются для отправки спама, что позволяет хакерам превращать машины своих жертв в

средства зарабатывания денег. На самом деле, по оценкам, более 80% всего спама происходит от бот-сетей, которые представляют собой сети скомпрометированных компьютеров, контролируемых одним злоумышленником для распределенных приложений. Даже когда ботнеты не задействованы, многие вирусы превращают своих хозяев в спам-боты, которые производят миллионы писем в день. Другие вирусы превращают свои хосты в открытые прокси, используемые спамерами для анонимной почты. Такие спам-письма сложнее обнаружить, поскольку они происходят от ботов, олицетворяющих законных пользователей.

Экономика спама

В конечном итоге причина спама продолжает насыщать почтовые ящики с нежелательной почтой, потому что она выгодна для спамеров. Чтобы проанализировать прибыльность спама, мы должны рассмотреть ряд факторов. Первоначальная стоимость, связанная с отправкой спама, – это расходы на ведение списков электронной почты, что может быть особенно значительным, если списки получены путем их покупки у других сторон. Отправка электронной почты несет небольшие расходы на отправителя, потому что почти все эксплуатационные расходы, связанные с хранением больших объемов информации, ложатся на получателей. Другие операционные расходы для спамеров могут включать в себя приобретение (или аренду) и поддержание бот-сетей и почтовых серверов. Наконец, риски, связанные с отправкой спама, включая уголовное преследование, должны быть учтены в модели, анализирующей экономику спама. Спам прибыльный, потому что общий доход обычно больше суммы этих расходов. Коэффициент конверсии зависит от процента получателей спама, которые следуют и выполняют какое-то желаемое действие, которое приводит к получению денег спамером. Это может быть, например, покупка продукта, подписка на услугу или просто щелчок на рекламе, которая может генерировать доход от рекламы для спамера. Коэффициент конверсии обычно чрезвычайно мал. Эксперимент, проведенный путем проникновения в ботнет, привел к 28 переходу из 350 миллионов сообщений, что обеспечило коэффициент конверсии 0,000008%. В целом, по оценкам исследователей, средний коэффициент конверсии для спама составляет менее

0,0001%. Тем не менее, несмотря на этот малый коэффициент оборачиваемости, огромное количество получателей позволяет спамерам восстанавливать свои расходы и быть прибыльным [48].

Черный список и серый список.

Один из самых популярных способов предотвращения спама от доступа к конечным пользователям – это черный список известных и подозреваемых источников спама и фильтрации входящей электронной почты на основе этих списков.

Несмотря на то, что для одного из интернет-провайдеров невозможно поддерживать точный черный список, существует несколько централизованных ресурсов, предназначенных для объединения списков источников спама, которые затем могут быть загружены почтовыми провайдерами для помощи в фильтрации спама. Черные списки спама часто публикуются с использованием системы доменных имен (DNS), и в этом случае они называются DNSBL (черные списки DNS). Они считаются противоречивыми, поскольку многие издатели DNSBL активно выступают против спама и агрессивно заносят в черный список, что потенциально препятствует доступу законных источников электронной почты к местам назначения. Сторонники утверждают, что агрессивный черный список может заставить интернет-провайдеров, которые терпят спамеров нести ответственность за их небрежность, в то время как противники обеспокоены потенциальным воздействием на свободу слова в Интернете.

Другая технология фильтрации спама, известная как серый список, вовлекает почтовый сервер получателя, первоначально отклоняющий почту от неизвестных отправителей. При получении электронной почты от неизвестного отправителя принимающий почтовый сервер отправляет отправителю сообщение «*временного отклонения*» и записывает соответствующую информацию. Поскольку это временное сообщение отклонения является стандартизированной частью протокола SMTP, законные отправители должны ответить путем повторной передачи отклоненного письма через определенный промежуток времени, после чего принимающий почтовый сервер примет это сообщение. Эта тактика основывается

на том, что спамеры обычно пытаются отправить электронную почту миллионам получателей и не имеют ресурсов для обработки этих временных отказов и повторных передач. Серый список обычно очень прост в настройке и не требует дальнейшего взаимодействия с администратором после его настройки. Хотя это по-прежнему соответствует протоколу SMTP, пользователи могут запросить немедленную почту, которая предотвращает использование серого списка. Тем не менее, это компромисс, который многие разработчики готовы сделать, особенно учитывая, насколько эффективно серый список уменьшает спам.

Фильтрация содержимого.

Последний механизм антиспама, который будет рассмотренн здесь, это сложная фильтрация контента. В этом методе сетевые администраторы развертывают приложения или расширения на почтовых серверах, которые анализируют текст и вложения каждого входящего письма, определяя вероятность того, что письмо будет спамом, и выполняют дальнейшие действия, основанные на этой оценке. *Наивная форма фильтрации* содержимого просто использует списки вложенных в черный список слов и помещает сообщение как спам, если оно содержит какое-либо из этих слов. Такая схема может обеспечить базовую защиту от спама, но обычно это приводит к большому числу ложных срабатываний, когда законные электронные письма относят к спаму, и к ложно-отрицательным решениям, когда спам-сообщения помечаются законными только потому, что они избегают спам-ключевых слов, например, используя замаскированные слова типа «V1agr@».

Для обеспечения лучших результатов были разработаны более сложные методы классификации адресов электронной почты на основе их содержимого. Один из наиболее эффективных методов известен как *байесовская фильтрация,* которая опирается на алгоритм машинного обучения, чтобы постепенно выяснять, как дифференцировать спам от законной электронной почты. Чтобы достичь этого «обучения», фильтр сначала подвергается периоду обучения, когда он просто записывает, считается ли электронное письмо спамом на основе ответов пользователей. Фильтр поддерживает список всех слов, найденных в содержимом

этих писем, и вычисляет вероятности того, что письмо, содержащее слово из списка, является спамом или законным. Как только эти вероятности будут откалиброваны в течение определенного периода времени, фильтр может присвоить рейтинг каждому входящему электронному письму, которое представляет вероятность его спама. Затем администратор установит пороговое значение, и, если в письме есть уровень спама выше этого порогового значения, предпринимаются соответствующие действия, такие как полное блокирование электронной почты или перенос ее в область карантина.

Недавние исследования в области фильтрации спама привели к ряду методов, которые направлены на использование совместной работы пользователей для кластеризации и блокировки спама. Однако в этой настройке необходимо следить за тем, чтобы вклад каждого пользователя не нарушал конфиденциальность чьих-либо электронных писем. Для достижения этой цели такие системы, как ALPACAS (A Large-scale, Privacy-Aware Collaborative Antispam System – крупномасштабная, конфиденциальная совместная антиспам-система), используют специально разработанную функцию преобразования, которая выполняется на каждом проверенном электронном письме для создания аналога «отпечатка пальца» для этого конкретного сообщения.

В идеальном случае было бы невозможно вычислить содержимое сообщения из его «отпечатка пальца», аналогично односторонней хэш-функции. Кроме того, методы уклонения, используемые спамерами, которые тонко изменяют содержимое каждого спам-сообщения, не должны влиять на его отпечаток. Было показано, что такие системы, как ALPACAS, более эффективны, чем традиционная байесовская фильтрация, и могут быть реализованы более широко в будущем.

5.3 Безопасность платежных систем и аукционов

5.3.1 Кредитные карты

Большинство онлайн-продаж происходит с использованием кредитных или дебетовых карт [3]. Операция онлайн-кредитной карты состоит из нескольких

этапов, в которых участвуют несколько сторон: клиент, банк клиента, называемый эмитентом, продавец, банк продавца, называемый приобретателем, и карточная сеть (например, MasterCard), называемая карточной ассоциацией.

На этапе авторизации клиент предоставляет продавцу расчетный счет кредитной карты вместе с дополнительной информацией, такой как дата истечения срока действия и код безопасности. Продавец утверждает транзакцию приобретателя, который передает его эмитенту через карточную ассоциацию.

Эмитент проверяет действительность карты и наличие средств на кредитной линии клиента. Если проверка завершается успешно, эмитент уменьшает кредитную линию клиента на сумму покупки и отправляет обратно продавцу разрешение транзакции через ассоциацию карт и приобретателя. Этап авторизации выполняется в режиме реального времени. Как только продавец получает разрешение на покупку, он отправляет покупателю приобретенные товары (рисунок 5.3).

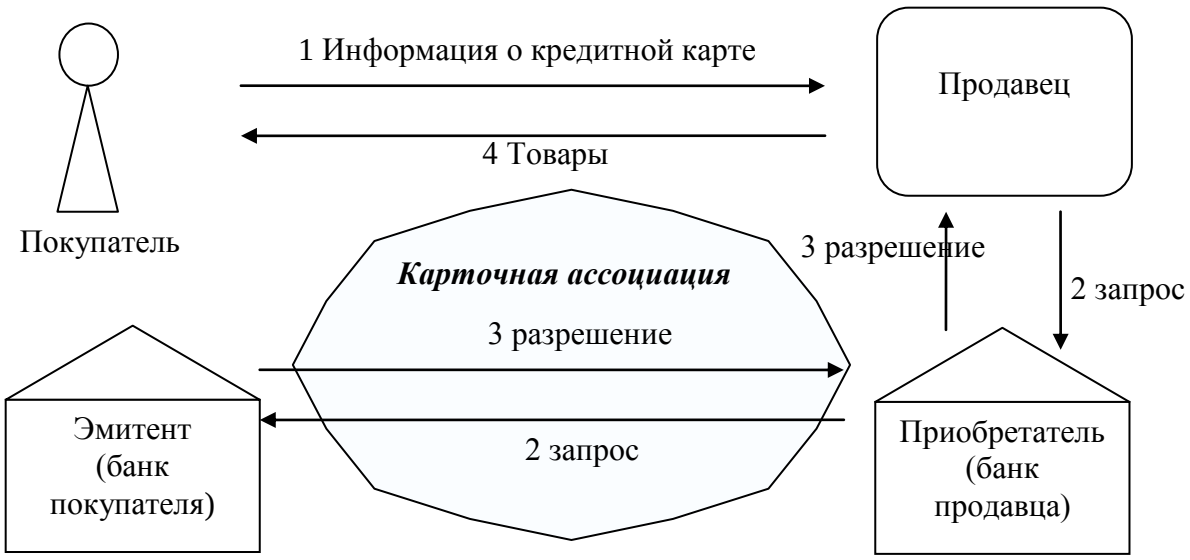


Рисунок 5.3 – Этап авторизации

Периодически, например, в конце каждого дня продавец отправляет приобретателю пакет транзакций для утверждения. Приобретатель перенаправляет их в карточную ассоциацию, которая обрабатывает расчет всех транзакций. В рамках расчета приобретателю зачисляется сумма покупки, и с эмитента списывается та же сумма. Как только урегулирование завершается, средства передаются от эмитента к приобретателю, продавец получит средства и покупатель

получит оплаченный счет. Урегулирование занимает от одного до трех дней с момента получения разрешений на доставку средств продавцу.

Мошенничество с кредитными картами и возврат платежей. Один из самых простых видов кредитного мошенничества исходит из того факта, что кредитные карты – это прежде всего физические объекты, которые представляют кого-либо в электронном мире – кредитная линия. В онлайн-транзакции по кредитной карте онлайн-идентификация, предоставляемая клиентом, – это номер кредитной карты и, возможно, идентификатор безопасности, оба из которых доступны при физическом доступе к карте. Если злоумышленник может получит эти цифры, он может совершать покупки с помощью кредитной карты жертвы. В результате, украденные номера кредитных карт стали обычным предметом торговли на черном рынке. Существует несколько защитных мер от мошенничества с кредитными картами и смягчения их последствий. Российский закон: 161-ФЗ защищает клиентов от мошеннических действий и несанкционированного использования карты. В статье 9 Федерального закона от 27.06.2011 № 161-ФЗ (ред. от 29.12.2014) «О национальной платежной системе» говорится, что оператор по переводу денежных средств обязан возместить сумму операции, совершенной без согласия клиента, если клиент уведомил его об этом не позднее дня, следующего за днём, в котором клиент получил уведомление от оператора по переводу денежных средств о совершении операции (уведомить обязан по закону). Особое внимание следует обратить на 15 пункт 9-й статьи этого закона: оператор по переводу денежных средств должен возместить сумму операции, совершенной без согласия клиента, если не докажет, что клиент нарушил порядок использования электронного средства платежа. Закон США ограничивает ответственность владельцев карт до 50 долларов США в случае мошенничества, независимо от того, сколько денег было потрачено. Этот закон защищает граждан от финансовых трудностей из-за мошенничества. Кроме того, каждый клиент имеет возможность инициировать возврат денег, если в своем платежном заявлении появляется мошенническая или некорректная покупка. В случае возврата платежа продавец получает возможность оспаривать иск, и в этом случае дело будет опосредовано как банками-продавцами, так и банками-клиентами.

Если возврат платежа не оспаривается или банк клиента побеждает в споре, деньги за транзакцию возвращаются клиенту, и продавец должен заплатить штраф за возврат. Важно отметить, что даже если продавец не несет ответственности за мошеннические платежи, как в случае кражи кредитной карты, он обязан возместить заказчику. Эта мера оказывает сильное давление на продавцов, чтобы проверить личность клиентов до авторизации покупки и защищает потребителей от финансовых трудностей в случае мошенничества.

Отмена карты. У эмитентов кредитных карт есть выделенные телефонные номера для отмены утерянной или украденной карты. После отмены карты все транзакции, использующие карту, отключаются. Кроме того, предпринимаются попытки совершения транзакций, чтобы помочь отслеживать злоупотребляющих лиц. Для дальнейшей защиты потребителей банки отслеживают *модели закупок клиентов* и применяют методы обнаружения мошенничества, чтобы определить вероятность того, что данная покупка является мошеннической. Индикаторы включают последовательные покупки в географически отдаленных регионах и суммы покупки намного больше, чем прошлые средние. В таких случаях банки обычно временно блокируют счет до тех пор, пока правомерность каждой сомнительной транзакции не будет подтверждена владельцем карты.

Аутентификация владельца карты Разработаны несколько методов для обеспечения дополнительного уровня безопасности поверх протокола авторизации кредитной карты. В системе 3D Secure, реализованной как MasterCard, так и Visa, держатель карты делится секретом с эмитентом и просит эмитента доказать владение этим секретом при попытке онлайн-покупки. В простейшей версии 3D Secure клиент регистрируется у эмитента, чтобы установить пароль, связанный с картой. Во время онлайн-покупки клиенту предлагается ввести пароль в веб-форму, которая появляется во всплывающем окне или фрейме, встроенном в страницу продавца. Эта веб-форма предоставляется эмитенту, а не продавцу. Пароль используется эмитентом в качестве доказательства того, что законный владелец карточки инициировал транзакцию. Хотя 3D Security нацелено на обеспечение дополнительного уровня предотвращения мошенничества, оно может запутать

клиента. Также, оно открывает дополнительную возможность для фишинговых атак, направленных на захват пароля владельца карты.

Еще один проблемный вопрос заключается в том, что банки могут использовать 3D Secure в качестве механизма для передачи ответственности клиенту в случае мошеннических транзакций.

Предоплаченные кредитные карты. Предоплаченные кредитные карты, также известные как карты с накопленной стоимостью, становятся все более популярной альтернативой традиционным кредитным и дебетовым картам. В отличие от кредитных карт, которые позволяют владельцам производить платежи по кредитным или дебетовым картам, которые связаны с банковским счетом, карты предоплаты инициализируются с указанным балансом перед их выдачей.

Этот баланс обычно не привязан к банковскому счету, и карточка может быть выдана физическому лицу или использоваться анонимно, в зависимости от эмитента карты. Поскольку для открытия счета не требуется кредитная линия или минимальный баланс, предоплаченные кредитные карты обычно используются несовершеннолетними.

В то время как предоплаченные карты могут быть удобны для использования в качестве альтернативы наличным деньгам, они часто предоставляют ограниченную защиту от мошенничества или не обеспечивают её вовсе из-за ограниченного потенциального воздействия мошенничества – вор может потратить только столько денег, сколько находится на балансе украденной карты.

5.3.2 Цифровые деньги

Цифровые деньги – это электронная валюта с той же анонимностью и отслеживаемостью как наличность физических денег. Операции с цифровыми деньгами включают плательщика, получателя платежа и, возможно, банка. Основная единица цифровых денег называется *электронной монетой* или, просто, монетой. Существует несколько целей безопасности, которым должны соответствовать схемы цифровых кассовых операций:

– *Конфиденциальность*. Электронные монеты не могут быть отнесены к плательщику или получателю, что отражает свойства, связанные с физическими деньгами.

– *Целостность*. Электронные монеты нельзя подделывать или дублировать, и соблюдаются законные транзакции.

– *Подотчетность*. Сделки не могут быть опровергнуты позднее, и споры по сделкам могут быть эффективно урегулированы. Легко гарантировать, что монеты могут быть созданы только с помощью достоверных источников. Для проверки подлинности монет продавцом может быть использована простая схема с цифровой подписью. Однако трудно обеспечить конфиденциальность, потому что банк может сопоставить снятие средств с последующими платежами. Для обеспечения конфиденциальности часто используются схемы *слепой подписи*, которые позволяют стороне, в данном случае банку, ставить цифровую подпись на сообщения без изучения содержимого самого сообщения. В простой схеме с использованием цифровых денег банк ставит слепую подпись на монеты, снятые клиентом. Получив монеты от клиента, продавец проверяет цифровую подпись и вносит монеты на депозит.

Во время такого обмена первый банк никогда не получает достаточной информации, чтобы связать этот конкретный вывод денег с их последующим депозитом. Предотвращение двойного расходования является более тонкой проблемой. Действительно, трудно помешать кому-то копировать электронные монеты и использовать их в более чем одном месте. В онлайн-системах двойные расходы могут быть предотвращены путем позволения банкам отзывать монеты, которые были потрачены, делая их недействительными. Для автономных систем одно решение зависит от подтвержденности личности для предотвращения двойных расходов. Каждая снятая монета содержит зашифрованную информацию о личности клиента, и каждая депонированная монета содержит зашифрованную информацию о личности торговца. С каждым депозитом раскрывается часть этой встроенной информации, поэтому один депозит не раскрывает никакой идентифицирующей информации. Однако последующие депозиты приводят к высокой вероятности

потери анонимности. Разработаны несколько криптографически защищенных схем цифровых денежных средств. Однако их практическое принятие было довольно ограниченным из-за отсутствия спонсорства со стороны правительств и финансовых корпораций, которые стремятся контролировать как можно больше денежных потоков.

Слепые подписи с RSA. Криптосистема RSA может использоваться для реализации простой схемы слепой подписи. Обозначив *открытый модуль* как n и закрытый ключ дешифрования как d, напомним, что сигнатура RSA на сообщении M задается как $s(M) = M^d \bmod n$.

Клиент выбирает случайный идентификатор монеты x и случайное число r, взаимно простое с n. Пара (x, r) представляет собой секретную монету.

Затем, используя открытый модуль n и открытый ключ зашифрования e, клиент вычисляет значение $y = r^e x \bmod n$ и отправляет его для подписания в банк. Обратите внимание, что банк не может получить идентификатор монеты x из значения y из-за «коэффициента ослепления». Предположим, что банк готов подписать значение y, предоставленное клиентом. Учитывая подпись $s(y)$ на y, клиент может получить подпись $s(x)$ по x, следующим образом:

$$s(x) = s(y)\, r^{-1} \bmod n,$$

где r^{-1} обозначает мультипликативную обратную от r по модулю n.

Чтобы гарантировать, что банк подписывает действительную монету, а не что-то еще, банк запрашивает от клиента генерацию k монет и предоставление криптографических хэшей для каждой из них. Банк случайно выбирает монету и подписывает ее. После этого, банк просит выявить оставшиеся k-1 монет. Затем банк проверяет, что каждая такая монета хешируется к значению, предоставленного ранее клиентом. Если проверка завершается успешно, монета, подписанная банком, действительна с вероятностью $1-1/k$.

5.3.3 Интернет-аукционы

Веб-сайты, такие как eBay.com, сделали онлайн-аукционы жизнеспособной бизнес-моделью для тех, кто продает отдельные товары и розничных компаний с

большими запасами. Онлайн-аукционы имеют много преимуществ перед традиционными продажами с фиксированной стоимостью и частными аукционами. Как и другие средства онлайн-продажи, онлайн-аукционы расширяют клиентскую базу до глобального рынка и позволяют мгновенно и легко обмениваться деньгами и товарами.

В частности, у аукционов есть дополнительные преимущества поощрения конкуренции между потребителями до тех пор, пока не будет определена самая высокая взаимно приемлемая цена. Тем не менее, анонимный характер Интернета представляет проблемы безопасности для онлайн-аукционов. Во-первых, поскольку любая сторона может зарегистрироваться как продавец, должен существовать механизм, позволяющий продавцам отвечать за мошенничество и кражу.

Сайты онлайн-аукционов обычно полагаются на системы репутации, чтобы обеспечить уверенность в легитимности продавцов. В этом случае клиенты имеют доступ к списку обзоров и рейтингов, связанных с каждым продавцом, и могут оценивать их по вопросам честного представления товаров, быстрой доставки и мошеннического поведения. По завершении транзакции каждому клиенту предлагается предоставить отзыв о продавце, чтобы позволить будущим клиентам оценивать честность, ответственность и профессионализм этого продавца.

Продавцы, которые нарушают правила, сразу же несут ответственность за эти нарушения посредством обратной связи с клиентами, а повторные нарушения могут привести к снижению продаж из-за низких рейтингов обратной связи, приостановки привилегий учетной записи или потенциальных судебных исков. Точно так же клиенты, которые вносят выигранные заявки на предметы, юридически обязаны завершить покупку этих предметов и несут ответственность за этот контракт. Покупатели, а также продавцы оцениваются и привлекаются к ответственности системой репутации. Однако следует соблюдать осторожность, чтобы не допустить, чтобы покупатели и продавцы давали свои оценки репутации за подкуп, например, за получение дополнительных услуг или платежей.

Еще одна проблема, связанная с онлайн-аукционами, – это торговля шиллингами, что является практикой того, когда коммерсант рекрутирует третьи

стороны для мошеннической заявки на один из перечисленных предметов этого продавца с целью раздувания текущей цены или предполагаемой желательности этого пункта.

В то время как на большинстве аукционных сайтов существуют строгие правила, запрещающие торговлю шиллингами, на самом деле трудно отличить ставки на шиллинге от законных ставок. Теория обнаружения ставок шиллингов (Shill) далека от точной науки, но ключевыми показателями могут быть использование вновь созданного аккаунта для размещения ставок, частых отклонений ставок, счетов, которые участвуют только в ограниченном пуле продавцов и отсутствии обратной связи с продавцами. На нескольких крупных аукционных сайтах используются сложные методы статистического вывода для определения ставок на шиллинги, но на момент написания таких методов такие методы часто хранятся в секрете, и детали этих алгоритмов обнаружения скрыты.

В работе «Inference of Online Auction Shills Using Dempster-Shafer Theory» [42] представлен подход обнаружения и проверки шиллинг-поведения для систем онлайн-аукционов на основе теории доказательств Демпстера-Шефера. Чтобы продемонстрировать осуществимость подхода, предоставлено исследование прецедентов с использованием реальных данных аукциона eBay. Результаты анализа показывают, что подход может обнаруживать шиллы и использование теории Демпстера-Шефера для объединения нескольких источников доказательств шиллинг-поведения может уменьшить количество ложноположительных результатов, полученных из одного источника данных.

5.4 Безопасность социальных сетей

Социальные сети предназначены прежде всего для использования онлайновыми сообществами для облегчения контакта между группами людей и людей с общими интересами или широким спектром особых интересов, начиная от датирования и заканчивая поиском работы с фотографией.

5.4.1 Социальные сети в качестве векторов атаки

Большим преимуществом социальных сетей, таких как Facebook, Instagram, LinkedIn и Twitter, является то, что они способствуют большому объему общения между людьми, которых называют «друзьями». К сожалению, эти повышенные уровни коммуникации и доверия также могут действовать как векторы атаки. Действительно, риски исходят из нескольких разных направлений. Во-первых, эти веб-сайты обычно предоставляют множество каналов связи между пользователями, в том числе возможность связаться с незнакомыми людьми, которые могут реально заниматься сбором информации. Риски таких контактов могут быть серьезными, поскольку компрометация учетной записи пользователя в социальных сетях может обеспечить доступ к частной информации, которая может быть использована для упрощения хищения личных данных, мошенничества или преследования. Этот риск еще больше увеличивается, поскольку исследования показывают, что до 15% пользователей социальных сетей будут отвечать взаимно запросам от незнакомца. Таким образом, даже если личная информация ограничена друзьями друзей, есть вероятность, что информация все равно может быть открыта для атаки, если друг отвечает на случайный запрос друга. Другой риск атаки для веб-сайтов социальных сетей исходит из того факта, что они являются высоко интерактивными, динамичными веб-приложениями. Например, несколько сайтов социальных сетей позволяют третьим лицам писать приложения, которые запускаются внутри домена безопасности сайта. Даже если база программного обеспечения для веб-сайта защищена, эти сторонние приложения являются потенциальными векторами атаки. Таким образом, администраторы веб-сайтов социальных сетей должны иметь строгие процессы проверки подлинности для сторонних приложений. Кроме того, поскольку они поддерживают различные виды интерактивной пользовательской коммуникации, веб-сайты социальных сетей являются потенциальными векторами для распределенных межсайтовых атак. Такие атаки могут использовать код, выполняемый в браузере жертвы, для распространения червей XSS, ссылок на вредоносные программы или спам-рекламы. Более того, поскольку пользователи

доверяют своим друзьям в социальных сетях, злоумышленники могут использовать это доверие для распространения вредоносного ПО или спама через скомпрометированные учетные записи. Такой компромисс может быть результатом фишинговой атаки, кражи данных из-за вредоносного ПО на машине жертвы или даже нарушения самой социальной сети.

5.4.2 Конфиденциальность

С ростом популярности веб-сайтов в социальных сетях люди чаще делают личную информацию общедоступной и видимой, по крайней мере, для некоторой части Интернета. Когда они собираются в группы, сайты социальных сетей часто позволяют не доверенным сторонам создавать достаточно полные профили человека. Например, работодателям нередко приходится искать личную информацию на сайтах социальных сетей для сбора дополнительных данных о потенциальных кандидатах на работу. Это нежелательное раскрытие личной информации может быть опасным, поскольку маленькие дети все чаще используют сайты социальных сетей. Интимные личные детали и механизм для установления контакта с незнакомыми людьми могут обеспечить легкий доступ для злоумышленников и мошенников. Из-за этих рисков сайты социальных сетей должны предпринять *три важных шага* для защиты конфиденциальности своих пользователей. *Во-первых,* пользователям должен быть предоставлен полный контроль над тем, какая личная информация доступна для сторон. Эти параметры должны быть легко доступны для пользователей и чрезвычайно просты в их редактировании, как пример системы, которая претерпела неоднократные изменения, чтобы упростить настройку для пользователей. Соответственно, пользователи несут определенную ответственность *за тщательное рассмотрение того*, в какой степени раскрывается их личная информация. *Во-вторых*, для параметров конфиденциальности должны быть назначены ограничения по умолчанию для защиты пользователей, которые не хотят или не могут настроить свои собственные настройки конфиденциальности. Например, сайты, разделяющие

личные данные, должны по умолчанию предоставлять данные только тем сторонам, с которыми пользователь явно инициировал контакт. Такие ограничения особенно важны для защиты маленьких детей, которые могут не знать об опасностях раскрытия очень важной личной информации в общедоступном Интернете или могут не иметь возможности правильно отредактировать свои собственные настройки. Наконец, сайты социальных сетей обязаны четко диктовать политику в отношении обмена пользовательской информацией. Пользователи должны знать, как их личную информацию можно получить и использовать третьими лицами. Например, сайт социальной сети Quechep подвергся суровой критике за автоматическую отправку без разрешения приглашений во всю электронную адресную книгу каждого пользователя. Другие менее авторитетные сайты заходят так далеко, что продают адреса электронной почты и личную информацию спамерам. Явные политики конфиденциальности позволяют пользователям размещать сайты социальных сетей, ответственные за эти действия.

Различные исследования показали, что сочетание друзей на веб-сайте социальной сети может содержать информацию, которая может дать возможность с некоторой степенью точности предсказать информацию о этом человеке, включая религию, расу, пол, возраст и сексуальную ориентацию. Таким образом, даже сочетание друзей может иметь последствия для конфиденциальности. Кроме того, исследования также показали, что можно *сопоставлять пользователей* между различными сайтами социальных сетей, просто сопоставляя списки друзей. Поэтому следует помнить о рисках, связанных с тем, что у одного и того же набора друзей есть сайт, который позволяет, по-видимому, анонимные псевдонимы для имен пользователей и сайт, на котором используется настоящее имя.

5.5 Безопасность SCADA

SCADA (Supervisory Control and Data Acquisition) – надзорный контроль и сбор данных. Сети SCADA – это системы контроля промышленных процессов и мониторинга распределенных систем.

Электрические сети, муниципальные системы водоснабжения и канализации, нефтепроводы и нефтеперерабатывающие заводы используют SCADA для проверки качества воды, делают соответствующие изменения в технологическом процессе или удаленно открывают и закрывают клапаны. Заводы используют SCADA для управления машинами, контроля потоков деталей и химических веществ и слежения за сбоями. В бизнес-зданиях используется SCADA для HVAC (отопление, вентиляция и кондиционирование воздуха) и механического контроля. Даже системы жизнеобеспечения в больницах можно рассматривать как системы SCADA [48].

Системы SCADA требуют защиты сами по себе, но кроме того должны быть защищены, поскольку компрометация системы SCADA может дать злоумышленнику преимущество перед корпоративными активами. Системы SCADA являются ИТ-системами и страдают от тех же многих уязвимостей, но имеют некоторые уникальные характеристики и особенности. Во-первых, SCADA-системы контролируют физический мир гораздо больше, чем типичное ПО. Система получает входные сигналы от датчиков, принимает решение на основе этих данных и затем управляет каким-либо механизмом исполнения принятого решения. Из-за этого системы SCADA являются более критичными, чем типичная ИТ-система. В большинстве случаев от отказа зависят человеческая жизнь или безопасность. Для SCADA требуется больше защиты, чем для обычного сервера. Во-вторых, системы SCADA построены из специального оборудования и ПО с гораздо более продолжительным жизненным циклом.

ИТ обычно обновляется на 3-5-летнем цикле, но системы SCADA часто имеют от 15 до 20 лет жизни. Оригинальный производитель системы может уже отсутствовать в бизнесе к моменту окончания установки. Это означает, что исправление систем SCADA может оказаться невозможным. Даже когда заплатки легко доступны, процесс может быть затруднен. Географическое распределение контроллеров и мониторов в сочетании с критичностью системы означает, что процесс должен быть тщательно спланирован в дни, когда приемлемо время простоя.

Существует множество конкретных *мер*, которые следует принять для обеспечения того, чтобы системы SCADA не подвергались риску. Выдержки из краткого перечня, разработанного Министерством энергетики США, приведены ниже, и многие другие можно найти в руководстве NIST 800-82.

1 Определить все подключения к сетям SCADA, включая старые модемные соединения и сотовые телефоны, затем отключить ненужные соединения.

2 Не полагаться на проприетарные протоколы для защиты установок SCADA.

3 Внедрить исправления и функции безопасности, предоставляемые поставщиками.

4 Физически защищать активы SCADA, сложную работу, учитывая географическое распределение частей системы.

5 Сосредоточиться на процессе управления рисками и управления изменениями в системе SCADA.

Поскольку компьютеры контролируют все больше умных домов, заводов и нефтеперерабатывающих заводов, безопасность SCADA становится еще более важной. Необходимо, чтобы вендоры были сосредоточены на стандартизированном аппаратном и программном обеспечении и повышали безопасность в будущих продуктах. В то же время руководителям и системным администраторам необходимо будет защитить существующие системы посредством применения внешних средств контроля.

5.6 Большие данные

Организации все чаще хранят и используют большие объемы данных. Также увеличилась потребность в обеспечении безопасности больших объемов данных в процессе их хранения и анализа. В свете этого нужно учитывать возможность потери этих стратегических ресурсов.

Каждый из них представляет отдельный набор вопросов в отношении обеспечения БИ. Хранилища данных, «Большие данные» и бизнес-аналитика

представляют собой взаимосвязанный набор систем, требующих оценки в рамках C.I.A.

Хранилище данных

Хранилище данных – это большой репозиторий данных и может иметь различные формы. Он будет включать данные, извлеченные из источника, которые хранятся и суммируются в форме, которая может быть использована приложениями для поддержки стратегического анализа. Исходные данные могут включать транзакционные записи, результаты бизнес-процесса и / или внешние данные. Хранилище данных выросло с корпоративного применения в 90-х годах в требования для большинства организаций. Независимо от подхода, хранилище данных служит корпоративной памятью и архивом.

Хорошо спроектированное хранилище данных – это хранилище данных, обеспечивающее выполнение функций извлечения, добычи и анализа данных для оценки производительности, добычи полезных сведений и поиска по ключевым словам. Безопасность в хранилище данных имеет жизненно важное значение, особенно для персональных данных, которые могут храниться в нем. В случае нарушения фирмам, возможно, придется заплатить за убытки клиентов, чьи записи были скомпрометированы [48].

Значение безопасности хранилища данных также может быть основано на потенциальной потере памяти организации. Кроме того, обеспечение того, чтобы сохраненные данные точно отражали события, которые хранятся, жизненно важно для памяти организации. Хотя это и не критично для краткосрочного восстановления, интегральный характер хранилища данных в большинстве организаций требует, чтобы было доступно восстановление бизнеса после катастрофы.

Большие данные

Текущей горячей темой для многих организаций является концепция больших данных (Big Data) и инструментов, которые поддерживают ее сбор и анализ. Несмотря на то что существует множество возможностей для бизнеса, чтобы успешно использовать постоянно растущее количество структурированных и

неструктурированных данных в организации, Big Data все еще находится в зачаточном состоянии, и необходимо учитывать пристрастия к инструментам, а также этические проблемы, связанные с использованием Big Data. К ним относятся защита доступа за пределами решений, скрытых за брандмауэрами и синтез того, как все эти решения будут работать вместе, и кто имеет к ним доступ.

Бизнес-аналитика

В контексте организации, приложения бизнес-аналитики публикуют результаты на основе данных, хранящихся в хранилище данных или средах с большими данными. Раскрытие критических данных является менее ключевым для инструментов бизнес-аналитики, хотя конкуренты могут найти ценность в показе данных организации, а скорее важнее надежность для принятия решений. По мере того, как инструменты становятся более внедренными в процесс принятия решений на всех уровнях организации, наибольшей угрозой может быть нарушение качества и целостности данных, на которых основываются решения.

Для устранения этой угрозы необходимы постоянная проверка и процесс обмена. Обеспечение прозрачности процессов принятия решений, включая прозрачность инструментов и используемых данных, сделает организацию более надежной, но, как и в случае с большинством вопросов в области ИТ, это может оказаться сложнее, чем кажется. Данные и способы их использования быстро меняются в каждой отрасли.

Ключевым также является учет формы и коллизий данных при перемещении из одной среды в другую. Возможна потеря информации при переходе, получен или потерян некоторый контекст. Понимание жизненного цикла и потока данных в организации обеспечит целостность и уверенность в том, что решения, принятые с использованием этих технологий, основаны на прочном фундаменте.

5.7 Облачная безопасность

Облачные сервисы – новое современное слово, и многие из обычных частных лиц имеют одну или несколько облачных сервисов хранения, музыки или

потокового видео, а также учетные записи электронной почты. Для предприятий облачные сервисы также расширяются, включая облачный веб-хостинг, все типы программного обеспечения как услуги, размещенную электронную почту и многое другое.

В рекомендация NIST SP 800-146 обсуждаются некоторые проблемы безопасности облачных вычислений.

3 Непреднамеренное раскрытие данных – необходимо позаботиться о хранении *конфиденциальных данных только в зашифрованном виде*

4 Конфиденциальность информаци – в облачной среде эта проблема осложняется распределенной природой облаков и возможным отсутствием осведомленности потребителей о том, где хранятся данные и кто имеет или может иметь доступ.

5 Целостность системы – ключевым атрибутом поддержания целостности облака является возможность разделения прав доступа для каждой группы пользователей при одновременном предотвращении злонамеренных атак. В облачной среде любое отсутствие видимости в облачных механизмах затрудняет проверку целостности приложений, размещенных в облаке.

6 Мульти-аренда. Существует риск сбоя изоляции, так как механизмы совместного использования объектов провайдера, зависят от сложных утилит, обеспечивающих изоляцию рабочих нагрузок потребителей. Укрепление уверенности в том, что логическое разделение является подходящей заменой физического разделения, является давней исследовательской проблемой, но эту проблему можно несколько смягчить, зашифровав данные перед их вводом в облако. Смягчение может происходить путем ограничения видов данных, которые обрабатываются в облаке, или путем заключения договоров с поставщиками для специализированных механизмов изоляции, таких как аренда целых компьютерных систем, а не виртуальных машин (моно-аренда), виртуальных частных сетей (VPN), сегментированных сетей или усовершенствованных средств управления доступом.

7 Браузеры. Многие облачные приложения используют браузер потребителя в качестве графического интерфейса, т.е. браузеры являются точками доступа к

облаку. К сожалению, браузеры являются сложными, скрывают недостатки безопасности и уязвимы почти в каждой проблеме общественной безопасности. Для повышения доверия к браузерам могут использоваться различные подходы, в том числе доступ к облакам из-за шлюза приложений или межсетевых экранов с фильтрацией сетевых пакетов, ограничение типов браузеров, одобренных для доступа к облаку, ограничение подключаемых модулей для браузеров, предоставляющих доступ к облаку, блокирование систем, которые обращаются к облакам через браузеры. Несмотря на практичность и полезность, большинство из этих методов увеличивают затраты, снижают функциональность или удобство.

8 Аппаратная поддержка – миграция виртуальных машин может ослабить цепочку доверия в доверенном платформенном модуле. Разные группы пытались виртуализировать Trusted Platform Module (TPM) реализующих сохранение набора контрольных сумм или создать аргумент, который может восстановить доверие повторно пробужденная виртуальная машина на другом оборудовании, но эта проблема остается открытой.

9 Управление ключами. Надлежащая защита пользовательских криптографических ключей, требует некоторого сотрудничества со стороны облачных провайдеров. Проблема заключается в том, что в отличие от выделенного оборудования, обнуление буфера памяти может не удалить ключ, если:

- память поддерживается гипервизором, который делает ее постоянной;
- виртуальная машина делает снимок, сделанный для целей восстановления;
- виртуальная машина сериализуется для перехода на другое оборудование.

В рекомендациях NIST SP 800-144 также обсуждаются вопросы безопасности для публичных облаков.

Несколько простых актуальны вопросов, которые нужно решить для начала работы с облаком.

Для частных лиц и предприятий первым шагом в облачной безопасности является обеспечение выбора надежного пароля и двухфакторная аутентификация, используемая при использовании сайта.

Затем следующий вопрос, особенно важный для предприятий, заключается в том, как шифруются данные: зашифрованы ли они перед отправкой в облачный сервис, зашифрованы ли они при транспортировке, хранятся ли в зашифрованном виде? Шифрование во всех этих случаях было бы лучшим сценарием.

Последний аспект шифрования касается того, кто владеет ключами шифрования. Для многих онлайн-сервисов ключи контролируются поставщиком услуг. Это означает, во-первых, что они могут расшифровать всю информацию, хранящуюся в онлайн-сервисе, даже если в соглашении об условиях обслуживания будет сказано, что они не имеют на это права.

Во-вторых, если их серверы будут скомпрометированы и ключи захвачены злоумышленниками, то все файлы станут доступными для чтения злоумышленниками. Принимая решение о том, следует ли доверять облачным провайдерам, стоит подумать, что почти каждый онлайн-сервис, а также личные данные были скомпрометированы в течение многих лет.

Остается открытым вопрос о том, как безопасно использовать криптографию из облака.

Как и в любой другой системе, облачные вычисления содержат уязвимости. Эти уязвимости при использовании злоумышленниками могут вызвать сбои в работе службы, потерю данных, кражу данных и т. д. Учитывая характер динамического совместного использования ресурсов в облаке, возможно, что классические атаки и уязвимости могут стать причиной большего вреда в облачной системе, если она не защищена должным образом.

Кроме того, динамическая и уникальная природа облака может требовать дополнительных мер, по сравнению с другими КС и эта природа также открывает двери для целого нового списка атак, которые можно использовать против облака.

Контрольные вопросы

1 Как протокол двухфазной фиксации помогает достичь целостности и доступности базы данных? Помогает ли это также в конфиденциальности и конфиденциальности?

2 Опишите проблемы автоматического распознавания компьютером CAPTCHA

3 Сравните преимущества внесения в черный список и серый список электронных писем?

4 Опишите основные различия между S / MIME и DKIM.

5 Что делать, если в выписке по вашей кредитной карте появляются покупки, которые вы не делали?

6 Какой механизм препятствует двойным использованиям цифровых денег? Является ли это эффективным сдерживающим фактором?

7 Назовите угрозы безопасности, которые возможны на сайтах социальных сетей.

8 Какие важные шаги должны предпринять сайты социальных сетей для защиты конфиденциальности своих пользователей?

9 В чем заключаются уникальные характеристики и особенности уязвимостей систем SCADA?

10 В чем заключаются проблемы безопасности больших данных?

11 Назовите особенности обеспечения безопасности облачных сред.

Список использованных источников

1 Астахова, Л.В. Теория информационной безопасности и методология защиты информации: учебное пособие / Л.В. Астахова. – Челябинск: Издательский центр ЮУрГУ, 2014. – 137 с.

2 Бернет, С.. Криптография. Официальное руководство RSA Security = RSA Security'sOfficialGuidetoCryptography / С. Бернет, С. Пэйн ; пер. с англ. под ред. А. И. Тихонова. – М. : Бином, 2009. – 382 с.

3 Внуков, А. А. Защита информации в банковских системах: учеб. пособие для бакалавриата и магистратуры / А. А. Внуков. – 2-е изд., испр. и доп. – М. : Издательство Юрайт, 2017. – 246 с. – (Серия : Бакалавр и магистр. Академический курс). – ISBN 978-5-534-01679-6.

4 Гайдамакин Н. А. Теоретические основы компьютерной безопасности / Н. А. Гайдамакин. – Екатеринбург, 2008. – 212 с.

5 ГОСТ Р ИСО/МЭК 27005-2010. Национальный стандарт Российской Федерации. Информационная технология. Методы и средства обеспечения безопасности. Менеджмент риска информационной безопасности" (утв. и введен в действие Приказом Росстандарта от 30.11.2010 № 632-ст) [Электронный ресурс]. – Режим доступа:

http://www.consultant.ru/cons/cgi/online.cgi?rnd=DCCAA914D514842E03A1A1679FE116E4&req=doc&base=OTN&n=666#01507836422327038

6 ГОСТ Р 50922-2006. Защита информации. Основные термины и определения. – Утв. Приказом Ростехрегулирования от 27.12.2006 № 373-ст. – Введ. 2008-02-01. – М.: Стандартинформ, 2008. – 8 с.

7 ГОСТ Р ИСО/МЭК 9594-8-98 ГОСУДАРСТВЕННЫЙ стандарт Российской Федерации. Информационная технология. Взаимосвязь открытых систем. Справочник. Ч а с т ь 8. Основы аутентификации (принят и введен в действие Постановлением Госстандарта России от 19.05.1998 № 215) / М.: ИПК Издательство стандартов, 1998. – 29 с.

8 ГОСТ Р 34.12-2015 Национальный стандарт Российской Федерации. Информационная технология. Криптографическая защита информации. Блочные шифры. (утв. и введен в действие Приказом Росстандарта от 19.06.2015 № 749-ст) / М.: Стандартинформ, 2016. – 12 с.

9 ГОСТ Р 34.10-2012. Национальный стандарт Российской Федерации. Информационная технология. Криптографическая защита информации. Процессы формирования и проверки электронной цифровой подписи" (утв. и введен в действие Приказом Росстандарта от 07.08.2012 № 215-ст) / М.: Стандартинформ, 2013. – 29 с. / на сайте Федерального агентства по техническому регулированию и метрологии [Электронный ресурс]. – Режим доступа: http://protect.gost.ru/document.aspx?control=7&id=180151

10 ДОКТРИНА информационной безопасности Российской Федерации. Утвержд. Указом Президента Российской Федерации от 5 декабря 2016 г. №646

11 Зима, В.М. Безопасность глобальных сетевых технологий / В.М. Зима, А.А. Молдовян, Н.А. Молдовян. – Спб.: БХВ-Петербург, 2000. – 320 с. – ISBN 5-94157-023-6

12 ИСО/МЭК ТО 13335-3:1998 Информационная технология. Рекомендации по управлению безопасностью информационных технологий. Часть 3. Методы управления безопасностью информационных технологий (Information technology – Guidelines for the management of IT Security – Part 3: Techniques for the management of IT security)

13 Казарин, О. В. Программно-аппаратные средства защиты информации. Защита программного обеспечения : учебник и практикум для вузов / О. В. Казарин, А. С. Забабурин. – М. : Издательство Юрайт, 2018. – 312 с. – (Серия : Специалист). – ISBN 978-5-9916-9043-0.

14 Краковский, Ю.М. Информационная безопасность и защита информации: учебное пособие / Ю.М. Краковский. – М.:ИКЦ «МарТ», Ростов н/Д: Издательский центр «МарТ», 2008. – 288с. – ISBN 978-5-241-00925-8

15 Лось, А. Б. Криптографические методы защиты информации : учебник для академического бакалавриата / А. Б. Лось, А. Ю. Нестеренко, М. И. Рожков. – 2-е изд., испр. – М. : Издательство Юрайт, 2018. – 473 с. – (Серия : Бакалавр. Академический курс). – ISBN 978-5-534-01530-0.

16 Лукацкий, А. Обнаружение атак/ А. Лукацкий . – СПб. [и др.] : БХВ – Санкт-Петербург, 2001. – 624 с.

17 Лясин, Д.Н. Защита информации (часть I): учебное пособие/ сост. Д.Н. Лясин, С.Г. Саньков, А.В. Степанова; ВПИ (филиал) ВолгГТУ. – Волгоград, 2016. – 98с.

18 Малюк А.А. Защита информации в информационном обществе: учебное пособие для вузов / А.А. Малюк. – М.: Горячая линия – Телеком, 2015. – 230 с.

19 Милославская, Н. Г. Управление рисками информационной безопасности / Н. Г. Милославская, М. Ю. Сенаторов, А. И. Толстой. – 2-е изд. – М.: Горячая линия-Телеком, 2014. – 130 с. – ISBN 978-5-9912-0362-3

20 Мельников, В. П. Защита информации: учебник для подготовки бакалавров по направлению 230100 "Информатика и вычислительная техника" / В. П. Мельников, А. И. Куприянов, А. Г. Схиртладзе; под ред. В. П. Мельникова. – Москва : Академия, 2014. – 297 с. – ISBN 978-5-4468-0332-3.

21 Мельников, В. П. Защита информации: учебное пособие для студ. Учреждений высш. Проф. Образования / В.П. Мельников, С.А. Клейменов, А.М. Петраков; под ред. С.А. Клейменова. – 6-е изд., стер. – М.: Изд. Центр «Академия», 2012. – 336 с. – ISBN 978-5-7695-9222-5

22 Мельников В.В. Защита информации в компьютерных системах В.В. / Мельников. – М.: Финансы и статистика; Электроинформ., 1997. – 368 с.

23 Методы спектрального анализа в задаче обнаружения аномалий информационных процессов телекоммуникационных сетей: монография / Н. А. Соловьев [и др.]; М-во образования и науки Рос. Федерации, Федер. гос. бюджет. образоват. учреждение высш. проф. образования "Оренбург. гос. ун-

т", Каф. прогр. обеспечения вычисл. техники и автоматизир. систем. – Оренбург : Университет, 2013. – 172 с. – ISBN 978-5-4417-0330-7.

24 О введении в системе Пенсионного фонда Российской Федерации криптографической защиты информации и электронной цифровой подписи: Постановление Правления ПФ РФ от 26.01.2001 № 15 [Электронный ресурс] / 6 с. // СПС КонсультантПлюс. – Режим доступа: http://www.consultant.ru

25 О безопасности: Федеральный закон от 28.12.2010 № 390-ФЗ (ред. от 05.10.2015) [Электронный ресурс] / "Собрание законодательства РФ", 03.01.2011, № 1, ст. 2. // СПС КонсультантПлюс. – Режим доступа: http://www.consultant.ru/document/cons_doc_LAW_108546/

26 Об информации, информационных технологиях и о защите информации: Федеральный закон от 27.07.2006 № 149-ФЗ (ред. от 18.12.2018) [Электронный ресурс] // СПС КонсультантПлюс // опубликован 18.12.2018 на официальном интернет-портале правовой информации. – Режим доступа: http://www.pravo.gov.ru

27 Об утверждении Перечня сведений конфиденциального характера: Указ Президента РФ от 06.03.1997 № 188 (ред. от 13.07.2015) [Электронный ресурс] // "Собрание законодательства РФ", 10.03.1997, № 10, ст. 1127 // СПС КонсультантПлюс. – Режим доступа: http://www.consultant.ru

28 Пушкарев В.П., Пушкарев В.В. Защита информационных процессов в компьютерных системах: Учебное пособие. – Томск: Томский межвузовский центр дистанционного образования, 2005. – 132 с.

29 Семененко, В. А. Программно-аппаратная защита информации: учеб. пособие для вузов / В. А. Семененко, Н. В. Федоров. – М. : МГИУ, 2007. – 340 с.

30 Смарт, Н. Криптография / Н. Смарт; пер. с англ. С. А. Кулешова; под ред. С. К. Ландо. – Москва: Техносфера, 2006. – 528 с.

31 Тишина, Н.А. Прикладные задачи безопасности информационно-телекоммуникационных систем: учебное пособие / Н.А.Тишина, Е.Н. Чернопрудова; Оренбургский гос. ун-т. – Оренбург: ОГУ, 2017. – 121 с.

32 Торстейнсон, П.. Криптография и безопасность в технологии . NET/ П. Торстейнсон, Г. А. Ганеш; пер. с англ. В. Д. Хорева ; под ред. С. М. Молявко. – М. : Бином, 2007. – 480 с.

33 ФСТЭК России. "Базовая модель угроз безопасности персональных данных при их обработке в информационных системах персональных данных" (Выписка) (утв. ФСТЭК РФ 15.02.2008) [Электронный ресурс] / 69 с. – Режим доступа: https://fstec.ru/tekhnicheskaya-zashchita-informatsii/dokumenty/114-spetsialnye-normativnye-dokumenty/379-bazovaya-model-ugroz-bezopasnosti-personalnykh-dannykh-pri-ikh-obrabotke-v-informatsionnykh-sistemakh-personalnykh-dannykh-vypiska-fstek-rossii-2008-god

34 ФСТЭК России. "Методический документ. Меры защиты информации в государственных информационных системах" (утв. ФСТЭК России 11.02.2014) / 176 с. [Электронный ресурс]. – Режим доступа: https://fstec.ru/component/attachments/download/675

35 ФСТЭК России. Методический документ ФСТЭК РОССИИ. Профиль защиты систем обнаружения вторжений уровня узла пятого класса защиты. – 6 марта 2012 г. – 67 с.

36 ФСТЭК России. Приказ ФСТЭК России от 14.03.2014 № 31 (ред. от 09.08.2018) "Об утверждении Требований к обеспечению защиты информации в автоматизированных системах управления производственными и технологическими процессами на критически важных объектах, потенциально опасных объектах, а также объектах, представляющих повышенную опасность для жизни и здоровья людей и для окружающей природной среды" (Зарегистрировано в Минюсте России 30.06.2014 № 32919) / опубликован на Официальном интернет-портале правовой информации http://www.pravo.gov.ru – 06.09.2018

37 ФСТЭК России. "Руководящий документ. Защита от несанкционированного доступа к информации. Термины и определения" (утв. Решением Гостехкомиссией России 30.03.1992) [Электронный ресурс]. – Режим доступа: / https://fstec.ru/component/attachments/download/298

38 Шаньгин, В. Ф. Защита компьютерной информации. Эффективные методы и средства: учеб. пособие для студентов вузов, обучающихся по направлению 230100 "Информатика и вычислительная техника" / В. Ф. Шаньгин. – М. : ДМК Пресс, 2008. – 544 с.

39 Шаньгин В. Ф. Комплексная защита информации в корпоративных системах: Учебное пособие / В.Ф. Шаньгин. – М.: ИД ФОРУМ: НИЦ ИНФРА-М, 2013. – 592 с. – ISBN 978-5-8199-0411-4.

40 Шнайер, Брюс. Прикладная криптография. Протоколы, алгоритмы, исходные программы на языке Си. М.: Триумф, 2012. – 816с.

41 Щеглов, А. Ю. Защита информации: основы теории : учебник для бакалавриата и магистратуры / А. Ю. Щеглов, К. А. Щеглов. – М. : Издательство Юрайт, 2018. – 309 с. – (Серия : Бакалавр и магистр. Академический курс). – ISBN 978-5-534-04732-5.

42 Dong F. Inference of Online Auction Shills Using Dempster-Shafer Theory [Электронный ресурс]/ F. Dong, Sol M. Shatz, H. Xu. – Режим доступа: https://www.researchgate.net/publication/224503697_Inference_of_Online_Auction_Shills_Using_Dempster-Shafer_Theory

43 Easttom, Chuck Computer Security Fundamentals / Chuck Easttom / Third Edition / Pearson Education, Inc., 2016. – 400 p. – ISBN-13: 978-0-7897-5746-3

44 Goodrich, Michael Introduction to Computer Security / Michael Goodrich, Roberto Tamassia /First Edition / Pearson Education Limited, 2014. – 514 pages. – ISBN-13 978-1-292-02540-7

45 Lo, C. A hybrid information security risk assessment procedure considering interdependences between controls / Lo C. C, Chen W. J. // Expert Systems with Applications, Vol. 39 . – 2012. – P.247-257.

46 Mutual Authentication [Электронный ресурс]. – Режим доступа: https://docs.oracle.com/cd/E19798-01/821-1841/bncbt/index.html

47 NIST. Engineering Principles for Information Technology Security (ABaseline for Achieving Security), Revision A.

48 Sagers, G. Information Technology Security Fundamentals / Glen Sagers Bryan Hosack / The Information Systems Collection Daniel J. Power, *Editor*. [Электронный ресурс]. – Режим доступа: Business Expert Press, LLC, 2016. – 157 pages. – ISBN-13: 978-1-60649-916-0

49 Snort: инструмент выявления сетевых атак [Электронный ресурс]. – Режим доступа: http://www.thg.ru/network/20051020/print.html

50 Yaruson/GostPlugin [Электронный ресурс]. – Режим доступа: https://github.com/yaruson/GostPlugin