Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Кафедра сетей и устройств телекоммуникаций

М. Н. Бобов, В. К. Конопелько

ОСНОВЫ АУТЕНТИФИКАЦИИ В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

Пособие по дисциплинам
«Защита программного обеспечения и баз данных
в сетях телекоммуникаций» и «Биометрические системы контроля
доступа в сетях телекоммуникаций» для студентов
телекоммуникационных специальностей 1-45 01 05 и 1-98 01 02
всех форм обучения

Минск БГУИР 2009

УДК 004.056:621.39(075.8)
ББК 32.973.26-018.2я73
Б72

Рецензенты:

доктор технических наук, профессор В. М. Колешко;

доктор технических наук, профессор В. Н. Булойчик

Бобов, М. Н.

Б72 Основы аутентификации в телекоммуникационных системах : пособие по дисц. «Защита программного обеспечения и баз данных в сетях телекоммуникаций» и «Биометрические системы контроля доступа в сетях телекоммуникаций» для студ. телекоммуникационных спец. 1-45 01 05 и 1-98 01 02 всех форм обуч. / М. Н. Бобов, В. К. Конопелько. – Минск : БГУИР, 2009. – 132 с. : ил.

ISBN 978-985-488-415-8

Данное издание является первым отечественным пособием по новому актуальному направлению в информационной безопасности – аутентификации пользователей в телекоммуникационных системах.

Изложены основы аутентификации в современных телекоммуникационных системах. Рассмотрены вопросы средств аутентификации, их эффективности, протоколы аутентификации, в том числе в Интернет, оценки их корректности.

Рекомендуется для студентов, магистрантов и аспирантов, обучающихся по телекоммуникационным специальностям, а также для специалистов в области прикладной математики, информатики и радиоэлектроники.

УДК 004.056:621.39(075.8)
ББК 32.973.26-018.2я73

ISBN 978-985-488-415-8

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Телекоммуникационные системы являются сложными системами, элементами которых являются субъекты системы, комплекс технических средств и обрабатываемая информация. Цель функционирования этих систем – обеспечение реализации информационных процессов с заданными характеристиками, направленных на обеспечение различных целевых задач пользователей систем. Для решения этих задач в системе осуществляются взаимодействия пользователей между собой, пользователей с серверами приложений и баз данных, серверов приложений и баз данных между собой. Краеугольным вопросом при организации данного взаимодействия является установление подлинности субъектов, пытающихся получить доступ к ресурсам системы.

Данная проблема решается путем использования средств и методов аутентификации, которые по месту приложения можно подразделить на локальные, прямые, опосредованные и автономные.

При локальной аутентификации вся система, включая механизм аутентификации и управления доступом, размещается внутри одного физического периметра безопасности. Владелец системы и (или) пользователь ведут и обновляют базу аутентификационных данных внутри этого периметра.

Прямая аутентификация предполагает непосредственное взаимодействие субъекта с устройством проверки подлинности при входе в систему. Системой могут коллективно пользоваться удаленным образом много различных пользователей. Механизмы аутентификации и контроля над доступом к системе по-прежнему размещаются внутри одного физического периметра. Владелец ведет и поддерживает в актуальном состоянии базу аутентификационных данных внутри каждой системы.

Непрямая аутентификация используется в современных сетевых серверных системах, содержит несколько точек обслуживания, которые требуют управления доступом и могут размещаться в различных местах. При необходимости пользователи обращаются к службам системы удаленным образом. Владелец ведет и поддерживает актуальной одну базу аутентификационных данных для всей системы.

Автономная аутентификация используется в системах с инфраструктурой открытого ключа, содержащих многочисленные автономные компоненты, которые способны принимать точные решения по управлению доступом даже в том случае, когда они не могут связываться с другими системами для получения авторитетных решений об аутентификации. Владелец соглашается с риском того, что такие решения могут иногда приниматься с использованием устаревших данных по управлению доступом или аутентификации, а значит, могут давать неправильные результаты.

По способу организации доступа используемые средства аутентификации разделяются на две группы: предназначенные для отдельных автономных компьютеров и для удаленного доступа. Очевидно, что локальная модель

связана с индивидуальным устройством. Модели прямой и непрямой аутентификации представляют собой различные стратегии для реализации удаленного доступа. Модель автономной аутентификации обеспечивает способ применения некоторых административных функций удаленной аутентификации в системах, которые не всегда могут установить удаленное соединение. Независимо от вида процесс определения подлинности включает два элемента: идентификацию и аутентификацию.

Идентификация – процесс распознавания элемента системы, обычно с помощью заранее определённого идентификатора или другой априорной информации. При идентификации происходит выбор элемента из множества.

Аутентификация – проверка истинности пользователя, процесса, устройства или другого компонента системы. При аутентификации происходит проверка подлинности заявленного идентификатора.

Являясь средствами защиты каналов доступа к телекоммуникационным системам, средства аутентификации должны обладать рядом специфических качеств, обеспечивающих надёжное закрытие этих каналов от несанкционированного доступа. Изучению принципов построения средств аутентификации и протоколов аутентификации, стойких по отношению к угрозам несанкционированного доступа, и посвящена эта книга. Она является первым отечественным пособием по новому актуальному направлению в информационной безопасности – аутентификации пользователей в телекоммуникационных системах.

Методам локальной аутентификации посвящены первые две главы пособия. В первой главе даётся общий обзор наиболее распространённых средств аутентификации и определяется обобщённый алгоритм их работы. Вторая глава посвящена практическим аспектам оценки и разработки эффективных средств аутентификации.

В трёх последующих главах рассматриваются вопросы аутентификации при удалённом доступе. В третьей главе описываются простые протоколы аутентификации, основанные на использовании паролей. В четвёртой главе рассматриваются основные протоколы аутентификации, используемые в сети Интернет. Пятая глава посвящена вопросам оценки стойкости протоколов аутентификации.

В основу пособия положены курсы лекций по дисциплине «Защита программного обеспечения и баз данных», читаемых на факультете телекоммуникаций БГУИР в течение ряда лет для студентов специальностей «Сети телекоммуникаций», «Защита информации в телекоммуникациях», «Системы распределения мультимедийной информации» по соответствующим учебным программам и учебным планам специальностей. Книга может быть полезна не только студентам, магистрантам, аспирантам и специалистам по телекоммуникациям, но и специалистам в других технических областях, поскольку все обозначенные выше проблемы носят общеметодологический характер.

ГЛАВА 1. СРЕДСТВА АУТЕНТИФИКАЦИИ ПОЛЬЗОВАТЕЛЕЙ

Реализация процедур аутентификации пользователей, которые включают в себя идентификацию и проверку подлинности, является общей проблемой для любых телекоммуникационных систем, в которых требуется обеспечивать разграничение доступа к обрабатываемой информации. Так как функционирование всех механизмов разграничения доступа, использующих аппаратные или программные средства, основано на предположении, что любой пользователь системы представляет собой конкретное лицо, то должен существовать некоторый механизм его опознания, обеспечивающий установление подлинности данного пользователя, обращающегося к системе.

Существуют три класса опознания (рис. 1.1), которые базируются:

а) на условных, заранее присваиваемых признаках (сведениях), известных субъекту (что знает субъект);

б) на физических средствах, действующих аналогично физическому ключу (что имеет субъект);

в) на индивидуальных характеристиках субъекта, его физических данных, позволяющих выделить его среди других лиц (что присуще субъекту).

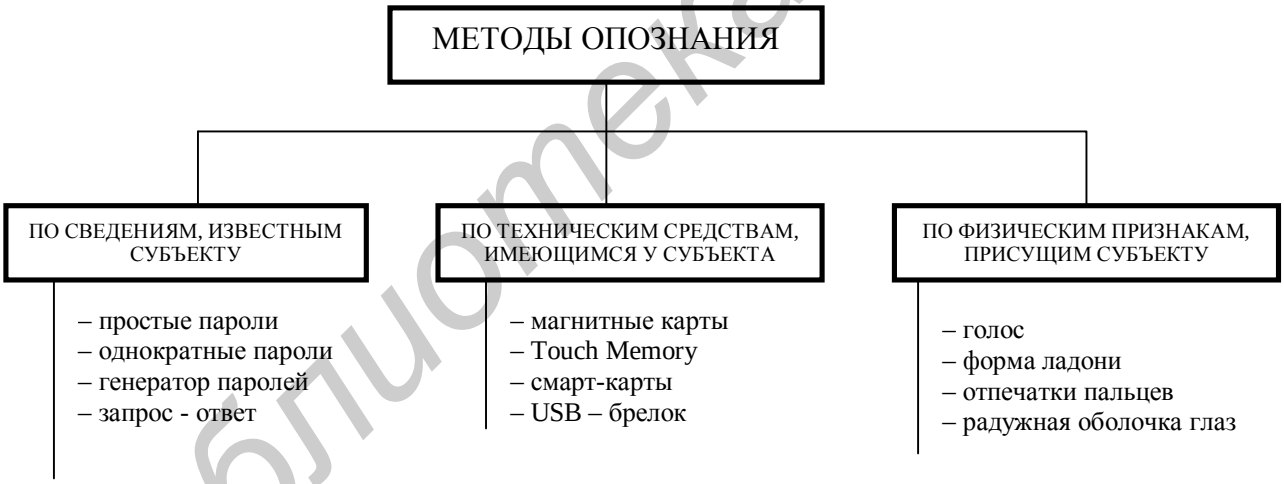

Рис. 1.1. Классификация методов опознания

1.1. Опознание на основе принципа «что знает субъект»

1.1.1. Метод паролей

Данный метод заключается в том, что пользователь на клавиатуре компьютера или специально имеющемся наборном поле набирает только ему известную комбинацию букв и цифр, которая собственно и является паролем. Введенный пароль сравнивается с эталонным, хранящимся в системе, и при положительном результате проверки пользователь получает доступ к системе. Приведенная схема опознания является простой с точки зрения реализации, так

как не требует никакой специальной аппаратуры и реализуется посредством небольшого объема программного обеспечения.

Рассмотрим алгоритм функционирования парольного средства аутентификации пользователей в операционной системе Microsoft Windows XP. Аутентификация пользователей в операционной системе Microsoft Windows XP основана на использовании паролей и реализуется следующими компонентами: Winlogon, GINA, LSASS, MSV1_0, SAM.

Winlogon – системный процесс, который отвечает за проведение операций входа и выхода пользователя в ОС.

GINA (Graphical Identification and Authentication) – файл динамической библиотеки, который предназначен для ввода имени пользователя и его пароля.

LSASS (Local Security Authentification SunSystem) – подсистема локальной аутентификации, которая управляет процессом аутентификации.

MSV1_0 – пакет аутентификации, который используется ОС при интерактивном входе пользователя. Предназначен для идентификации и аутентификации пользователя.

SAM (Security Account Manager) – объект, который ведет базу данных имен пользователей и паролей.

Схема взаимодействия компонентов ОС Microsoft Windows XP в процессе интерактивного входа пользователя представлена на рис. 1.2.

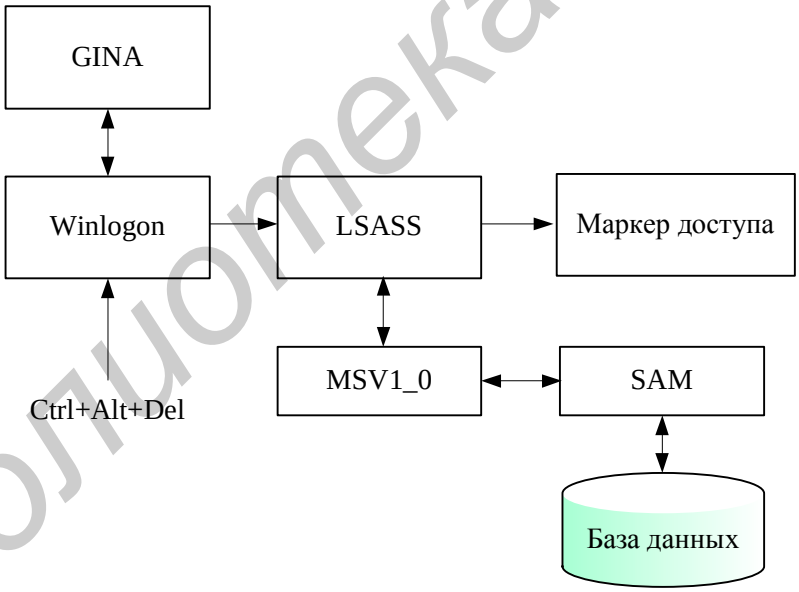

Рис. 1.2. Компоненты, участвующие в процессе интерактивного входа пользователя в операционную систему Microsoft Windows XP

Алгоритм функционирования средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP при интерактивном входе представлен на рис. 1.3. Процесс аутентификации включает в себя следующие этапы.

А. Запрос на вход в систему.
Б. Ввод имени и пароля.
В. Идентификация пользователя.

Г. Аутентификация пользователя.

Д. Создание маркера доступа.

А. *Запрос на вход в систему*. Пользователь нажимает комбинацию клавиш Ctrl+Alt+Del. В результате этого Winlogon вызывает GINA, который выводит на экран поля, необходимые для ввода имени и пароля пользователя.

Б. *Ввод имени и пароля*. После набора пользователем имени и пароля GINA передает эти данные в Winlogon, который производит хеширование пароля, создает уникальный локальный идентификатор защиты (SID – Security IDentifer) для этого пользователя и вызывает LSASS.

В. *Идентификация пользователя*. LSASS подключает пакет аутентификации MSV1_0, который принимает от Winlogon имя пользователя и хешированную версию пароля и посылает в SAM запрос на получение из учетной записи пользователя, которая хранится в базе данных SAM, хешированного пароля.

Идентификация заключается в нахождении введенного пользователем имени в базе данных SAM. Если введенное пользователем имя не содержится в базе данных SAM, то MSV1_0 возвращает в LSASS статус отказа.

Г. *Аутентификация пользователя*. В случае нахождения имени пользователя в базе данных MSV1_0 сравнивает хешированный пароль пользователя с тем, который хранится в базе данных SAM и соответствует учетной записи пользователя. Если эти данные совпадают, MSV1_0 генерирует локально-уникальный идентификатор сеанса входа (LUID – Locally Unique IDentifer) и передает его вместе с SID в LSASS. Если данные не совпадают, то MSV1_0 возвращает в LSASS статус отказа.

Д. *Создание маркера доступа*. Собрав необходимую информацию, LSASS вызывает исполнительную систему для создания маркера доступа. Исполнительная система создает маркер доступа для интерактивного сеанса, который включает в себя SID пользователя. После успешного создания маркера доступа LSASS дублирует его, создавая описатель, который передается Winlogon, а свой описатель закрывает. На этом этапе LSASS сообщает Winlogon об успешном входе. При наличии в LSASS статуса отказа Winlogon сообщает пользователю о неправильно введенном имени или пароле. Программа Winlogon дает пользователю несколько попыток ввода правильных идентификатора и пароля. После превышения числа допустимых попыток программа прекращает свое выполнение.

Схема с использованием простого пароля имеет два недостатка:

– большинству пользователей сложно запомнить произвольное число, используемое в качестве пароля;

– пароль может быть использован другим лицом, так как его легко подсмотреть.

Модернизацией схемы с использованием простого пароля является пароль однократного использования. В этой схеме пользователю выдается список из N паролей, такие же N паролей хранятся в системе. Данная схема обеспечивает большую степень безопасности, но она является и более сложной.

Здесь при каждом обращении к системе синхронно используется пароль с текущим номером, а все пароли с предыдущими номерами вычеркиваются. В случае если старый пароль из предыдущего сеанса стал известен другому пользователю, система его не воспринимает, так как действующим будет следующий по списку пароль.

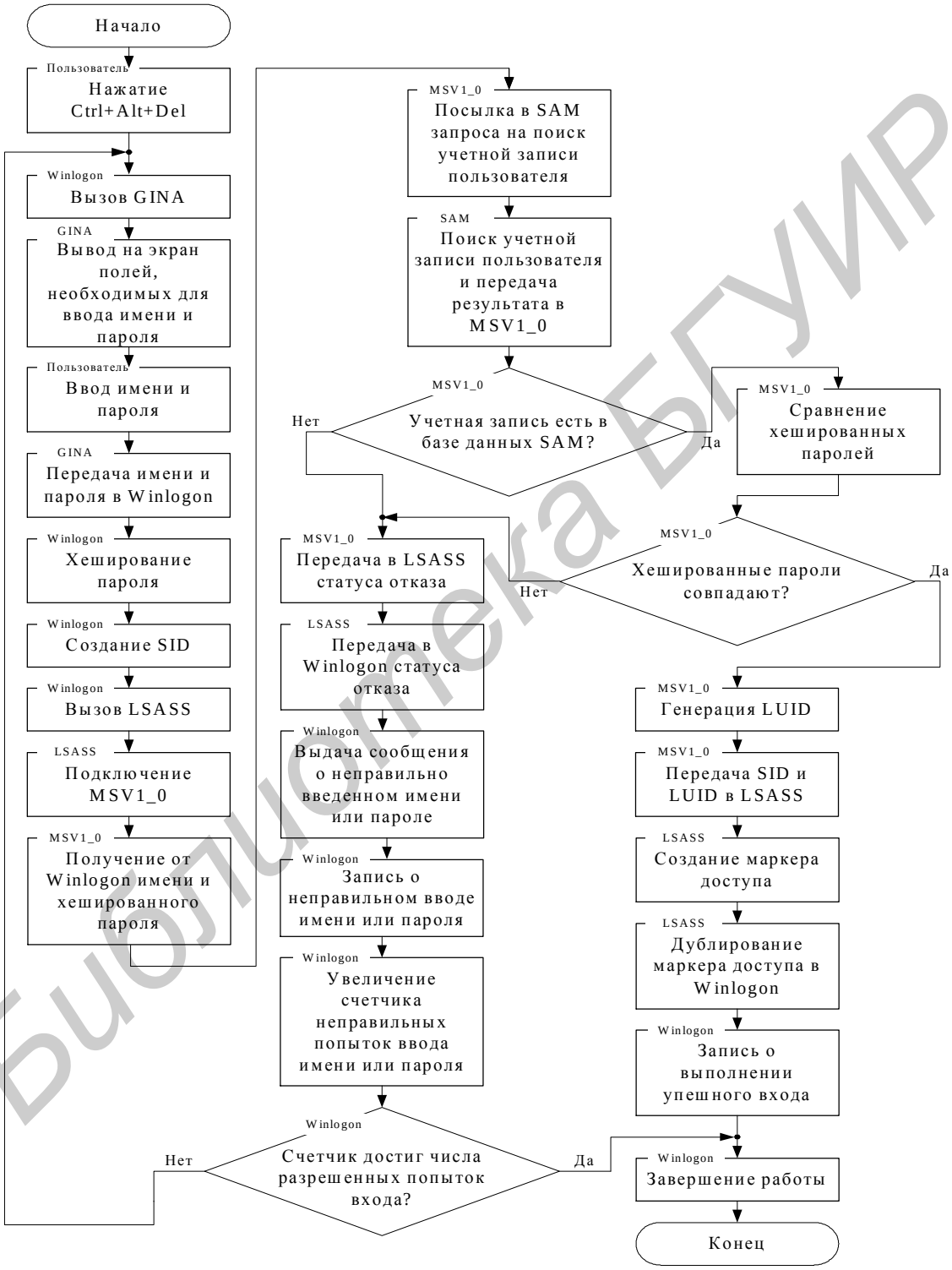

Рис. 1.3. Алгоритм функционирования средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

Схема паролей однократного использования имеет следующие недостатки:

– пользователь должен помнить или иметь при себе весь список паролей и следить за текущим паролем;

– в случае если встречается ошибка в процессе передачи, трудно определить, следует ли передавать тот же самый пароль или послать следующий;

– необходимо иметь разные таблицы паролей для каждого пользователя, так как может произойти рассинхронизация работы.

Последний недостаток можно устранить, используя генератор паролей. В этом случае в ЭВМ реализуется алгоритм, осуществляющий преобразование

$$F(x, k) = y,$$

где x, k, y – двоичные векторы соответственно характеристического номера, ключа и пароля.

Реализация процедуры опознания пользователя сводится к двум задачам: заготовке паролей и установлению подлинности.

При заготовке паролей с помощью преобразования $F(x, k) = y$ получают набор чисел

$$X_i^j, \Pi_i^j,$$

где i – номер пользователя; j – номер обращения данного пользователя;

Π – текущее значение пароля, сформированное на ключе k.

Сгенерированный набор чисел выдается соответствующим пользователям.

Опознание системой пользователя I происходит следующим образом: пользователь с номером i вводит парольный набор X_i^j, Π_i^j в ЭВМ. Программа опознания выделяет номер пользователя X_i^j, а также запоминает пароль Π_i^j. Для каждого i-го пользователя существует свой счетчик обращений S_i. В случае, если $j \le S_i$, программа выдает сообщение о несанкционированном доступе (НСД). В противном случае включается генератор паролей. Преобразование $F(x_i^j, k)$ на действующем ключе k выдает число y, которое сравнивается с паролем Π_i^j. В случае совпадения y и Π_i^j пользователь считается опознанным, а в случае несовпадения выдается сигнал о несанкционированном доступе.

Использование генератора паролей избавляет от необходимости хранить таблицы паролей для каждого пользователя, однако первые два недостатка при его использовании сохраняются.

1.1.2. Метод «запрос-ответ»

В методе «запрос-ответ» набор ответов на m стандартных и n ориентированных на пользователя вопросов хранится в ЭВМ и управляет программой опознания. Когда пользователь делает попытку включиться в работу, программа опознания случайным образом выбирает и задает ему некоторые (или все) из этих вопросов. Пользователь должен дать правильный ответ на все вопросы, чтобы получить разрешение на доступ к системе. Вопросы могут быть выбраны таким образом, чтобы пользователь запомнил ответы и не записывал их.

Модификация этого метода предполагает изменение каждый раз одного или более вопросов, на которые пользователь давал ответ до этого.

Существует два варианта использования метода «запрос-ответ», вытекающих из условий $m = 0$ или $n = 0$. Вариант с $m = 0$ предполагает, что вопросы составлены на основе различных фактов биографии индивидуального пользователя, представляют собой имена его друзей, дальних родственников, старые адреса и т.д. Пользователь, который сам предложил опознавательный вопрос, всегда даст на него правильный ответ, чего не сможет сделать злоумышленник. Иногда предпочтительнее вариант с $n = 0$, т.е. пользователям задается большее количество стандартных вопросов и от них требуются ответы на те, которые они сами выберут. Достоинство рассмотренной схемы в том, что пользователь может выбирать вопросы, а это дает весьма высокую степень безопасности в процессе включения в работу. В то же время нет необходимости хранить в системе тексты вопросов для каждого пользователя, достаточно хранить указатели на вопросы, выбранные данным пользователем, вместе с информацией, устанавливающей его подлинность. Текст каждого стандартного вопроса необходимо ввести для хранения только один раз, поэтому в системе с большим числом пользователей это может дать экономию памяти.

Наряду с достоинствами метод «запрос-ответ» все же имеет и недостатки, ограничивающие возможность его использования, а именно:

– метод требует проявления изобретательности от самих пользователей, что для них является дополнительной нагрузкой;

– большинство людей, как правило, предлагают стереотипные вопросы и ответы в качестве опознавательных, поэтому весьма вероятно, что настойчивый нарушитель может, собрав статистику, предугадать многие вопросы и ответы;

– процедура обмена множеством опознавательных запросов и соответствующих им ответов может быть сложной и утомительной для пользователей;

– метод «запрос-ответ» может использоваться только для небольших организованных групп пользователей, он неприменим для массового использования в силу некоторой громоздкости.

1.2. Опознание на основе принципа «что имеет субъект»

К данному классу опознания относятся методы, основывающиеся на физических средствах, которые имеет при себе данный пользователь, обращающийся к системе. К ним относятся магнитные карточки, смарт-карты, USB-ключи, таблетки Touch Memory и прочие подобные средства, которые можно объединить общим названием – *электронный ключ*.

Электронным ключом в самом общем смысле являются физические носители идентификатора субъекта и его пароля. Кроме того, на носителях содержится дополнительная информация, необходимая в процессе опознания субъекта.

Для восприятия смарт-карта должна иметь ридер. В процессе обмена информацией с ридером происходит опознание смарт-карты. Опознание субъекта происходит после подтверждения им того, что именно он является владельцем смарт-карты в результате ввода с клавиатуры PIN-кода.

Аналогом ридера для USB-ключей выступает стандартный USB-порт, а для электронного ключа Touch Memory – считывающее устройство.

1.2.1. Идентификационные магнитные карты

В магнитных картах информация записывается на нескольких дорожках магнитного слоя и представляет собой данные, используемые для идентификации. К этим данным относятся: номер пользователя или его имя, пароль, количество допустимых использований карты и т.д. Наряду с очевидной простотой использования магнитные карты обладают низкой защищенностью от копирования содержимого. Для защиты от копирования магнитные карты снабжаются различными защитными средствами. Один из методов состоит в нанесении магнитного слоя обычного типа поверх второго слоя с более высокой коэрцитивной силой, т.е. для изменения состояния первого слоя требуется более сильное магнитное поле. В этом случае обычными методами невозможно считать или изменить запись нижнего слоя. Считывающее устройство, читая карту, содержащую идентификатор, вначале создает поле, стирающее любую запись, сделанную обычным способом, а затем уже считывает лежащую ниже «твердую» запись, в которой находится идентификационная информация.

В другом методе используется постоянная магнитная разметка ленты, которая наносится в процессе ее производства. Метод, известный под названием «влажной разметки», состоит в определенной ориентации осей ферромагнитных кристаллов до момента, пока наполнитель еще не высох, причем селективная ориентация осей кристаллов в различных частях ленты создает магнитную запись, которую никак нельзя изменить. Чтобы прочесть эту запись, кристаллы необходимо подвергнуть воздействию постоянного магнитного поля с определенной ориентацией. Изменение положения кристаллов вдоль ленты будет наводить внешнее поле, которое можно прочитать с помощью обычных, удобно расположенных головок. Изготовленные таким образом идентификационные карточки могут обеспечить «уникальную» идентичность, которую трудно подделать, поскольку для этого требуется овладеть технологией производства магнитных покрытий и влажной разметки.

1.2.2. Электронные ключи

Электронный ключ в самом общем смысле представляет собой физический носитель секретного кода, являющегося аутентификатором пользователя. В отличие от парольных систем использование электронного ключа (ЭК) имеет ряд преимуществ:

– пользователю не надо запоминать значение пароля, так как пароль записан в ключе;

– пользователь освобожден от проблемы защиты пароля от компрометации при его вводе, так как пароль считывается из ключа;

– все функции по защите от подделки пароля или его несанкционированного использования (метод разовых паролей, метод «рукопожатия») возлагаются на электронный ключ;

– секретный код можно сделать сколь угодно большим, так как пользователь с ним непосредственно не работает.

Рассмотрим алгоритм функционирования средства аутентификации с использованием смарт-карт. Средство аутентификации с использованием смарт-карты реализуется следующими модулями: центральный процессор (ЦП), ПЗУ, ОЗУ, ЭСППЗУ, программное обеспечение (ПО), дисплей, клавиатура, приемо-передатчик.

ЦП предназначен для реализации криптографических алгоритмов и разграничения доступа к хранящейся в памяти смарт-карты информации. В ПЗУ хранится исполняемый код ЦП, а ОЗУ используется в качестве рабочей памяти. Энергонезависимая память для хранения информации пользователя смарт-карты (ЭСППЗУ) необходима для хранения изменяемых данных владельца карты. ПО предназначено для осуществления взаимодействия смарт-карты с рабочей станцией. Приемопередатчик предназначен для приема и передачи информации как от смарт-карты к рабочей станции, так и наоборот.

Схема взаимодействия компонентов, участвующих в процессе аутентификации пользователя с использованием смарт-карты, представлена на рис. 1.4.

Алгоритм функционирования средства аутентификации с использованием смарт-карты представлен на рис. 1.5 и включает в себя следующие этапы.

А. Ввод смарт-карты в специальное устройство для чтения.

Б. Идентификация смарт-карты.

В. Аутентификация пользователя.

Г. Формирование записи о результате входа в систему.

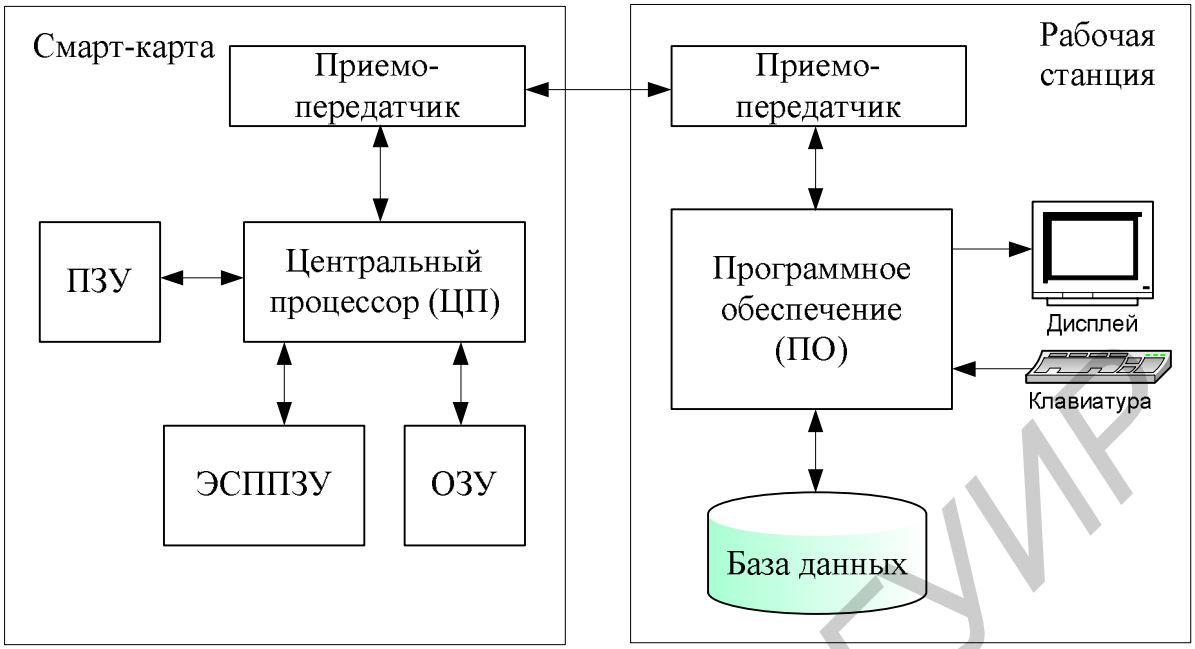

Рис. 1.4. Компоненты, участвующие в процессе аутентификации пользователя с использованием смарт-карты

А. *Ввод смарт-карты в специальное устройство для чтения и ввод пользователем своего PIN-кода.* Пользователь вставляет смарт-карту в специальное устройство для чтения смарт-карт (ридер, терминал), которое подключено к рабочей станции. ПО рабочей станции посылает в смарт-карту управляющий сигнал и в ЦП смарт-карты загружается исполняемый код из ПЗУ смарт-карты.

Затем ПО выдает на монитор запрос на ввод пользователем своего PIN-кода. Пользователь вводит с клавиатуры свой PIN-код, который поступает в ПО рабочей станции. Эталонный PIN-код владельца смарт-карты в зашифрованном виде хранится в ЭСППЗУ смарт-карты.

Б. *Идентификация смарт-карты.* ПО посылает запрос ЦП смарт-карты на выдачу персональной информации, которая содержит срок окончания работы смарт-карты и ее серийный номер.

Если срок окончания работы смарт-карты подошел к концу, то ПО выдает на монитор сообщение о том, что смарт-карта устарела и пользователю необходимо изъять ее.

Если срок работы смарт-карты еще не истек, то ПО ищет в базе данных учетную запись с полученным серийным номером смарт-карты. Идентификация смарт-карты считается успешной, если ПО находит в базе данных учетную запись с таким серийным номером.

Если на предъявленный серийный номер учетной записи нет, то это означает, что смарт-карта не является зарегистрированной в данной системе и, следовательно, не проходит идентификацию. В таком случае ПО выводит на монитор сообщение о том, что формат смарт-карты является неверным и предлагает пользователю изъять ее.

В. *Аутентификация пользователя.*

В1. *Выработка ПО случайного числа.* ПО вырабатывает случайное число и посылает его в ЦП смарт-карты. Случайное число записывается в ОЗУ смарт-карты.

В2. *Вычисление смарт-картой хеш-кода.* ЦП смарт-карты вычисляет хеш-код от зашифрованного на общем для смарт-карты и ЭВМ ключе PIN-кода, сцепленного со случайным числом и ключом приложения. Полученный хеш-код ЦП отправляет в ЭВМ.

В3. *Вычисление устройством доступа хеш-кода.* ПО вычисляет хеш-код от зашифрованного на общем для смарт-карты и ЭВМ ключе введенного пользователем PIN-кода, сцепленного со случайным числом и ключом приложения.

В4. *Сравнение результатов устройством доступа и принятие решения о подлинности пользователя.* ПО сравнивает вычисленные хеш-коды. Если хеш-коды не совпадают, то ПО выдает на монитор сообщение о том, что введен неверный PIN-код и предлагает повторить попытку ввода PIN-кода. Так как число попыток ввода ограничено, то если PIN-код введен неверно установленное количество раз, устройство доступа блокирует смарт-карту. Если хеш-коды совпадают, то ПО переходит к записи результата входа в систему.

Г. *Формирование записи о результате входа в систему.* При совпадении хеш-кодов ПО делает запись об успешном входе в систему и выводит на монитор соответствующее сообщение. Если хеш-коды не совпадают, то ПО выдает запись об отказе в доступе.

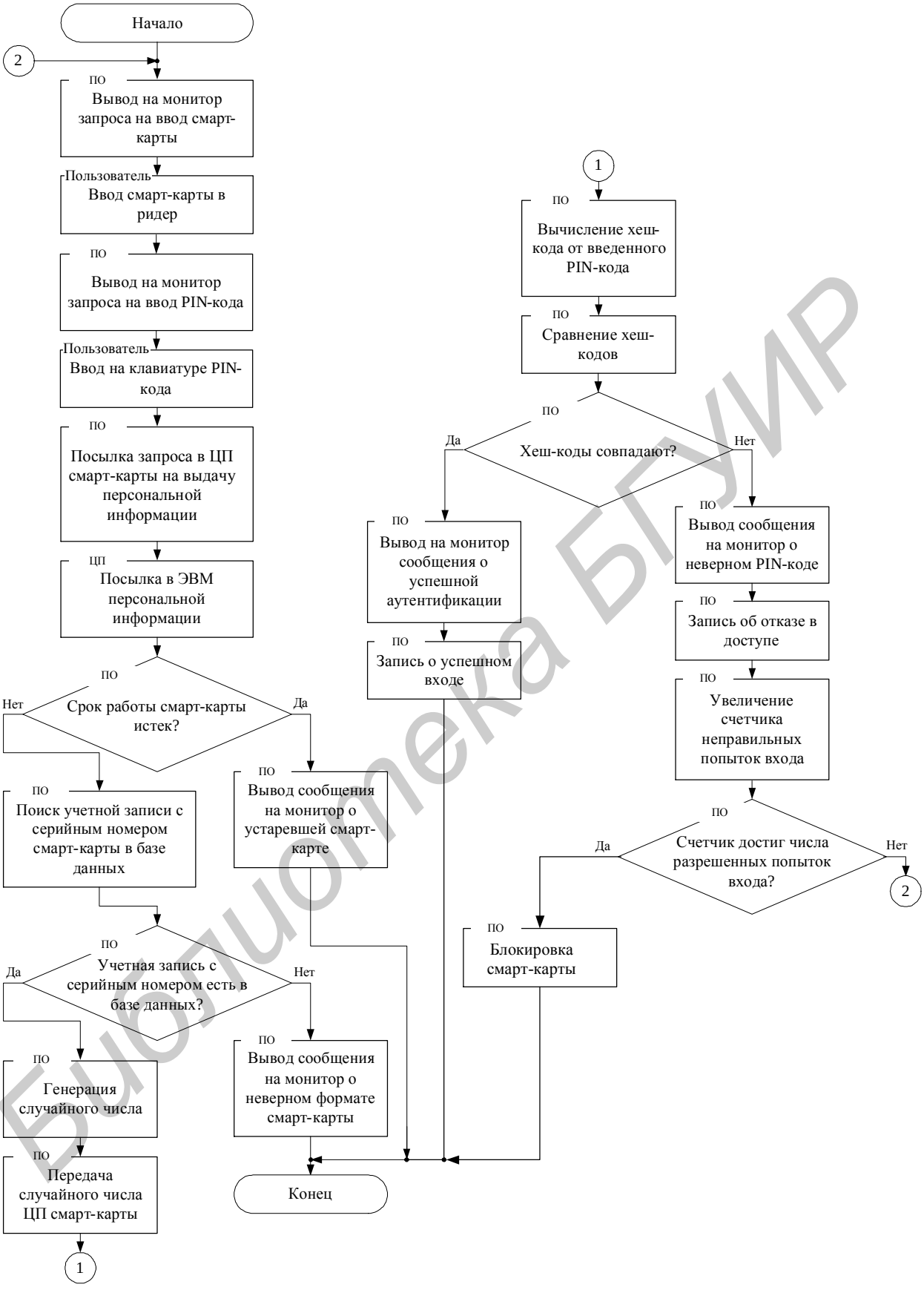

Рис. 1.5. Алгоритм функционирования средства аутентификации с использованием смарт-карт

1.3. Опознание на основе принципа «что присуще субъекту»

Данный принцип опознания базируется на определении индивидуальных характеристик, присущих каждому пользователю и позволяющих выделить его среди других лиц. К наиболее широко используемым персональным характеристикам относятся голос, личная подпись, форма ладони и отпечатки пальцев. В последнее время появилось еще несколько методов физического опознания – по структуре сетчатки глаз, сопротивлению определенных участков кожи, запаху тела и др. В каждом случае способ опознания состоит в измерении индивидуальных характеристик и вычислении индексов, аналогичных характеристическим параметрам распознавания образов, которые можно передать в центральную ЭВМ для сопоставления с набором индексов, хранящихся в памяти ЭВМ и взятых непосредственно у интересующего лица.

1.3.1. Параметры идентификации физиологических признаков

Механизм опознания личной подписи может измерять число касаний и отрывов пера от бумаги, среднюю вертикальную скорость движения пера, число вертикальных отклонений и множество других подобных параметров. Эти характеристики могут быть самыми разнообразными, однако не все из них являются независимыми, и задача состоит в том, чтобы выбрать хороший набор характеристик с достаточно малой взаимной корреляцией. Проверка подлинности подписи зависит от движения пера, которое нельзя воспроизвести по виду подписи, зафиксированной на бумаге. Это практически полностью исключает возможность подлога, так как умение профессионально подделывать подписи, основано на внешнем виде почерка. Набор измеряемых характеристик должен сохраняться в тайне, так как их знание может привести к подделке подписи посредством тренировки в копировании измеряемых характеристик. Как показала практика, обеспечение секретности – это сложная задача.

Аналогичные особенности характерны и для других методов опознания этого класса. Например, некоторые устройства, определяющие форму ладони, измеряют прозрачность тканей кожи между пальцами для защиты от подлогов с помощью картонных шаблонов. Механизмы, построенные на анализе отпечатков пальцев, используют мельчайшие детали в виде разветвлений, окончаний и пробелов в линиях на кончиках пальцев. Так как в каждом отпечатке содержится множество таких отличий, измеряемые характеристики могут базироваться на выбранном наборе деталей. Существуют два основополагающих алгоритма распознавания отпечатков пальцев:

- по отдельным деталям (характерным точкам);
- по рельефу всей поверхности пальца.

В первом случае устройство регистрирует только некоторые участки, уникальные для конкретного отпечатка, и определяет их взаимное расположение. Во втором случае обрабатывается изображение всего отпечатка.

Метод опознания субъекта *по лицу* основан на уникальности черт лица. Метод заключается в преобразовании черт конкретного лица в алгоритмическую

модель, которая сравнивается или с фотографией на пропуске, или с содержимым базы фотографических данных.

Метод опознания субъекта *по радужной оболочке глаза* основан на уникальности рисунка радужной оболочки каждого субъекта. Радужная оболочка субъекта сканируется, разворачивается и преобразуется в цифровую последовательность. Подтверждение подлинности субъекта происходит на основании сравнения полученной цифровой последовательности с эталонной.

Метод опознания *по образцу голоса* основан на том, что у каждого субъекта неповторимый голосовой рисунок, который определяется полом, физическими особенностями субъекта, в частности его речевым аппаратом: типом строения голосовых связок, полостью носа, формой рта, таких характеристик голоса, как частота и амплитуда. Этот метод построен на выделении различных сочетаний частотных и статистических характеристик голоса.

1.3.2. Средство аутентификации с устройством сканирования отпечатка пальца

Данное устройство использует отпечаток пальца в качестве биометрического признака личности и реализуется такими компонентами, как датчик изображения папиллярных линий кожи пальца, USB-разъем, USB-порт, программное обеспечение (ПО), монитор, клавиатура.

Датчик изображения папиллярных линий кожи пальца (ДИПЛКП) предназначен для сканирования отпечатка пальца, преобразования полученного изображения в цифровую форму и передачу его на USB-разъем. USB-разъем и USB-порт служат для передачи цифровой информации от датчика изображения папиллярных линий кожи пальца в ЭВМ. ПО предназначено для работы с изображением отпечатка пальца, сравнения отпечатка пальца с эталонным, хранящимся в базе данных, и для управления диалогом с пользователем.

Схема взаимодействия компонентов, участвующих в процессе аутентификации пользователя по отпечатку пальца, представлена на рис. 1.6.

Алгоритм функционирования средства аутентификации по отпечатку пальца представлен на рис. 1.7 и включает в себя следующие этапы.

А. Ввод имени пользователя.
Б. Сканирование отпечатка пальца.
В. Работа с файлом отпечатка пальца.
Г. Идентификация пользователя.
Д. Аутентификация пользователя.
Е. Принятие окончательного решения.

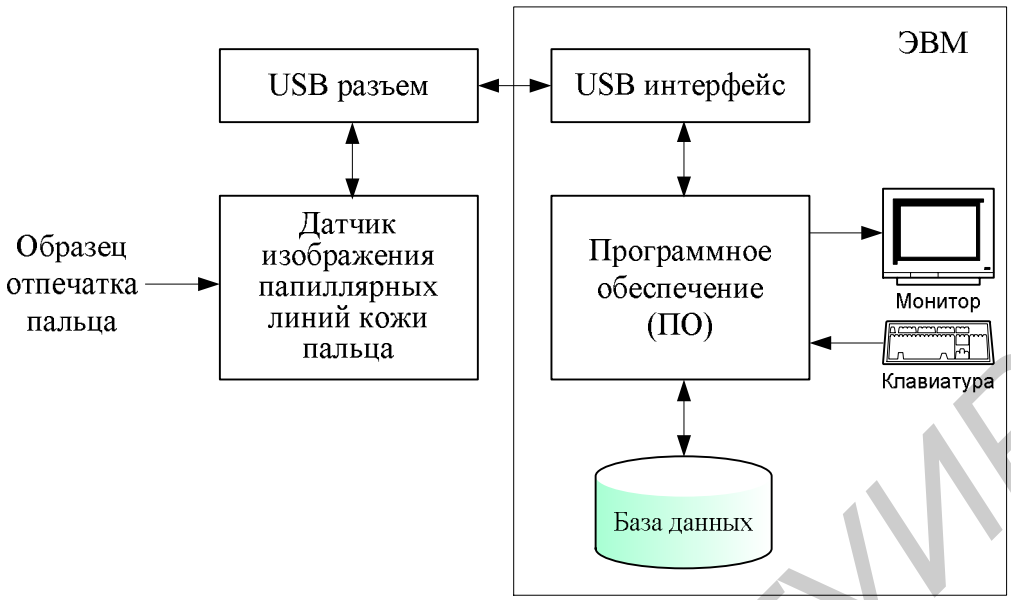

Рис. 1.6. Компоненты, участвующие в процессе аутентификации пользователя по отпечатку пальца

А. *Ввод имени пользователя.* Для входа в систему пользователь запускает на ЭВМ процесс аутентификации. ПО средства аутентификации выдает на монитор запрос на ввод пользователем своего имени. Пользователь вводит имя с клавиатуры.

Б. *Сканирование отпечатка пальца.* После ввода пользователем своего имени ПО активирует датчик изображения папиллярных линий кожи пальца. После этого пользователь прикладывает свой палец к сканирующей области датчика, который сканирует отпечаток пальца пользователя и преобразует его в цифровую форму.

В. *Работа с файлом отпечатка пальца.*

В1. *Формирование файла картинки отпечатка пальца.* От датчика изображения папиллярных линий кожи пальца через USB-разъем в USB-порт цифровая форма отпечатка пальца попадает в ЭВМ. ПО сохраняет полученную от датчика информацию в файл-образ отпечатка пальца с заданными разрешением, числом пикселов на дюйм, количеством уровней яркости. Далее ПО создает образ папиллярных линий пальца, где темным участкам соответствуют выступы папиллярного рисунка, а светлым – впадины.

В2. *Улучшение качества исходного изображения отпечатка.* Для улучшения структуры гребней папиллярных линий и резкости их границ ПО производит низкочастотную фильтрацию изображения отпечатка пальца.

В3. *Бинаризация изображения отпечатка.* ПО производит пороговую обработку изображения отпечатка пальца, в результате которой пиксели изображения, цвета которых меньше заданного порога делаются чёрными, а те, цвета которых выше, – белыми.

В4. *Утончение линий изображения отпечатка.* ПО производит утончение линий изображения отпечатка пальца до тех пор, пока эти линии не станут равными одному пикселу.

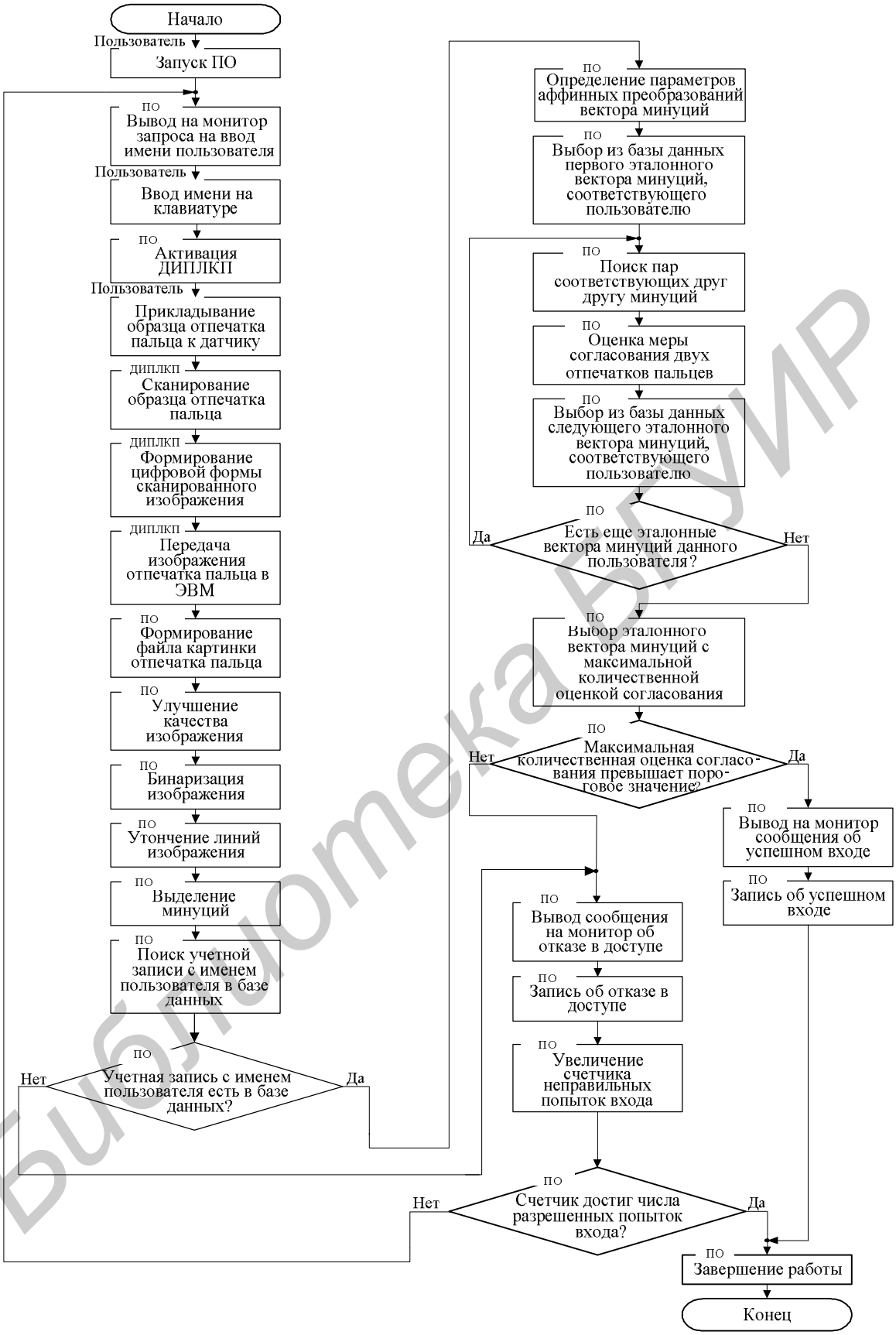

Рис. 1.7. Алгоритм функционирования средства аутентификации с устройством сканирования отпечатка пальца

Г. *Идентификация пользователя.* ПО ищет в базе данных учетную запись с введенным именем. Идентификация считается успешной, если ПО находит в базе данных учетную запись с введенным именем пользователя.

Д. *Аутентификация пользователя.*

Д1. *Выделение минуций.* ПО производит локальную обработку всего изображения отпечатка пальца с помощью маски 9×9 пикселов и подсчета числа пикселов, находящихся вокруг центра маски и имеющих ненулевые значения. Пиксел в центре маски принимается за минуцию, если он сам имеет ненулевое значение и если число «соседей» также ненулевое и равно 1 или 2.

Координаты обнаруженных минуций, а также углы их ориентации ПО записывает в вектор минуций.

Д2. *Регистрация данных.* При положительном результате идентификации ПО выбирает эталонный вектор минуций, соответствующий данному пользователю и определяет параметры аффинных преобразований, при которых некоторая минуция сформированного вектора будет согласована с некоторой минуцией эталонного вектора.

При отрицательном результате идентификации ПО выводит на монитор сообщение об отказе в доступе. Количество попыток ограниченно. После исчерпания всех попыток ПО закрывается, а его запуск блокируется.

Д3. *Поиск пар соответствующих друг другу минуций.* На каждом шаге ПО подвергает аффинным преобразованиям координаты минуций из полученного вектора и полученные новые координаты сопоставляет с каждой из координат минуций эталонного вектора.

Д4. *Оценка меры согласования двух сопоставляемых отпечатков.* ПО осуществляет количественную оценку согласования двух сопоставляемых отпечатков, как отношение квадрата количества найденных пар минуций к произведению количества минуций в полученном векторе минуций на количество минуций в эталонном векторе минуций, умноженное на сто процентов.

Е. *Принятие окончательного решения.* В базе данных хранятся несколько эталонных векторов минуций одного и того же отпечатка пальца, полученных при разных условиях его сканирования. ПО сравнивает полученный вектор минуций с каждым из эталонных векторов. После сравнений ПО выбирает тот эталонный вектор минуций, количественная оценка согласования которого максимальна. Если эта количественная оценка согласования превышает некоторое пороговое значение, то ПО вырабатывает положительный результат аутентификации пользователя и выдает на монитор сообщение об успешной аутентификации. В противном случае ПО вырабатывает отрицательный результат и выдает на монитор сообщение об отказе в доступе.

1.3.3. Алгоритм функционирования средства аутентификации с устройством распознавания голоса

Аутентификация основана на использовании образца голоса в качестве биометрического признака и реализуется следующими компонентами: микрофоном, программным обеспечением (ПО), монитором, клавиатурой.

Микрофон служит для ввода образца голоса, а клавиатура – для ввода имени пользователя. ПО предназначено для работы с образцом голоса пользователя, для выделения и сравнения векторов речевых признаков и для управления диалогом с пользователем. На монитор выводятся необходимые пользователю сообщения.

Схема взаимодействия компонентов, участвующих в процессе аутентификации пользователя по образцу голоса, представлена на рис. 1.8.

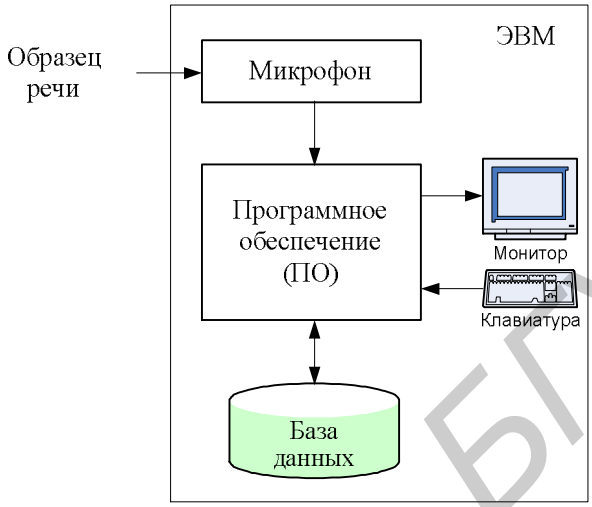

Рис. 1.8. Компоненты, участвующие в процессе аутентификации пользователя по образцу голоса

Алгоритм функционирования средства аутентификации по образцу голоса представлен на рисунке 1.9 и включает в себя следующие этапы.

А. Ввод имени пользователя и образца голоса.

Б. Идентификация пользователя.

В. Выделение векторов речевых признаков.

Г. Принятие окончательного решения об аутентификации.

А. *Ввод имени пользователя и образца голоса*. Пользователь запускает пользовательский интерфейс, который предлагает ввести свое имя и образец голоса. Затем пользователь набирает на клавиатуре свое имя и подает на микрофон фрагмент речи, который представляет собой голосовой пароль.

Б. *Идентификация пользователя*. ПО производит поиск в базе данных учетной записи с введенным именем. Идентификация считается успешной, если ПО находит в базе данных учетную запись с введенным именем пользователя.

В. *Выделение векторов речевых признаков*. Поданный на микрофон образец голоса записывается в память ЭВМ и обрабатывается ПО, которое определяет векторы речевых признаков, представляющие характерные параметры входного речевого сигнала. Векторы речевых признаков определяются с помощью линейного предсказания для нахождения его кепстральных коэффициентов. ПО генерирует векторы речевых признаков в виде кепстральных коэффициентов методом векторного квантования и формирует из них матрицу.

Далее производится вычисление мер близости между сгенерированной матрицей кепстральных коэффициентов и каждой из трех эталонных матриц, хранящихся в базе данных и соответствующих имени пользователя. В результате ПО принимает решение о том, превышает мера близости пороговое значение или нет.

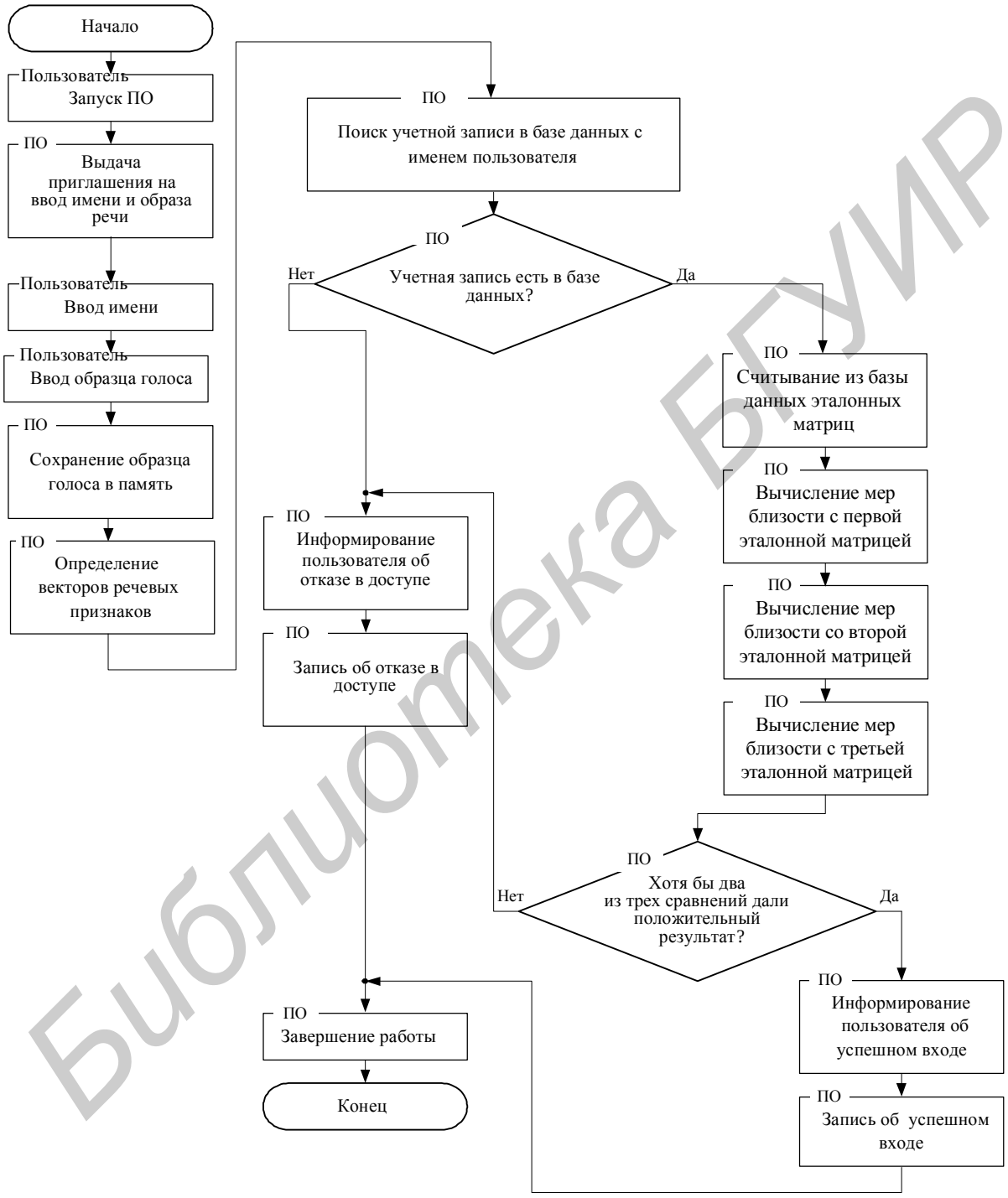

Рис. 1.9. Алгоритм функционирования средства аутентификации по образцу голоса

Г. *Принятие окончательного решения об аутентификации*. Если меры близости между сгенерированной матрицей и хотя бы двумя из трех эталонных

матриц не превышают порогового значения, то ПО принимает положительное решение об аутентификации пользователя и выводит соответствующие сообщение на монитор. В противном случае ПО принимает отрицательное решение об аутентификации пользователя и выводит на монитор сообщение об отказе в доступе.

1.3.4. Особенности опознания по физическим признакам

Значения характеристик, получаемые в процессе работы средств опознания по физическим признакам, всегда имеют разброс в небольшой области с некоторым вероятностным распределением, поэтому для принятия решения об аутентификации необходимо определить «окно приемлемости» для каждого параметра. При экспериментальной оптимизации качества механизма опознания размер такого «окна» может меняться, но он всегда остается больше некоторого минимума, так как практически не существует абсолютно надежного набора параметров для опознания по физиологическим признакам. Если взять слишком широкое значение «окон», то система примет любой запрос, а если очень узкое, то на все попытки аутентификации запросов последует отказ, в том числе и на запросы законных пользователей. На рис. 1.10 представлены типичные кривые, показывающие соотношение между ошибками этих двух типов.

На практике, как видно из рис. 1.10, оба типа ошибок нельзя свести к нулю одновременно независимо от величины установленного порога. В этом коренное отличие последнего класса систем опознания от первых двух, где опознание считается установленным только после абсолютного совпадения предъявленного аутентификатора.

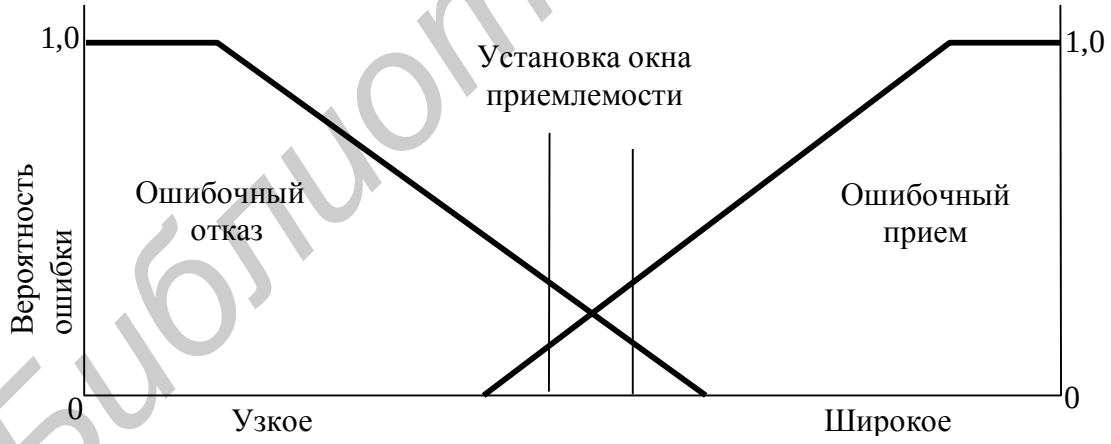

Рис. 1.10. Два вида ошибок при опознании по индивидуальным характеристикам

В системе опознания по физиологическим признакам всегда обязательно наличие процедуры распознавания образов. От качества реализации этой процедуры напрямую зависит качество опознания. Система опознания собирает от пользователей набор значений параметров, который служит в качестве образца, и определяет положения и размеры «окон». В нее может быть заложена возможность совершенствования в процессе накопления опыта

опознания пользователей. Если выбирается слишком высокое значение «окон», то возможен прием любого запроса, а если очень узкое, то на все попытки последует отказ, в том числе и на запросы законных пользователей. Целью процедуры распознавания образов является сведение к минимуму ошибок обоих типов. Процент ошибочных отказов можно уменьшить, если предоставить пользователю возможность выполнить процедуру идентификации более одного раза и принять успех любой из попыток как достаточное условие подтверждения личности. Ясно, что вместе с тем возрастет и вероятность ошибочного опознания. Использование метода опознания по физическим признакам при опознании лица, не известного заранее, является практически неосуществимым, так как это требует выработки критериев оценки персональных характеристик, чтобы выделить одного индивидуума среди всех других, обслуживаемых данной системой. Хотя физические параметры содержат достаточную информацию для опознания отдельного лица, эта процедура основывается на очень тщательном изучении хорошо измеренных параметров и выделении всех их особенностей. При практической же реализации данного метода опознание осуществляется на основе значительно меньшего объема информации и служит только для проверки характеристик одного, предположительно известного лица посредством сопоставления с соответствующей записью, хранящейся в памяти ЭВМ и составленной на основе ранее выполненных измерений его характеристик.

Таким образом, методы опознания, основанные на определении характеристик личности пользователя, более сложны и дороги при реализации, чем методы, основанные на использовании паролей и физических ключей, так как, во-первых, необходимо осуществлять довольно сложную процедуру распознавания и сравнения образов. Во-вторых, в них значительно более вероятен отказ в доступе действительному пользователю из-за ошибок самой системы, а необходимость сбора характеристик и установления подлинности пользователя до того, как он обратится к системе, делает эти методы неудобными и малопригодными для распределенных систем с большим количеством пользователей.

1.4. Функциональная структура средства аутентификации

Анализ реализаций средств аутентификации, приведенных в предыдущих разделах, показывает, что они имеют общие закономерности функционирования. Каждый из рассмотренных алгоритмов работы средств аутентификации содержит этапы:

– обработки входных воздействий и преобразования их в необходимый вид;

– идентификации и аутентификации;

– принятия решения о разрешении доступа к защищаемой системе или его запрете;

– контроля исполнения управляющего воздействия.

Таким образом, в процессе работы алгоритма каждое из средств аутентификации субъекта выполняет следующие функции:

- обнаружения;
- опознания;
- управления;
- контроля.

К функции *обнаружения* относятся те элементы алгоритма работы средства аутентификации, которые обеспечивают выявление подлежащих анализу входных воздействий и их преобразование в форму, необходимую для работы средства аутентификации. К функции обнаружения, например, можно отнести процессы сканирования отпечатка пальца и обработки файла рисунка отпечатка пальца.

К функции *опознания* относятся те элементы алгоритма работы средства аутентификации, которые обеспечивают проверку законности субъекта и устанавливают, является ли он тем, за кого себя выдает. К функции опознания, например, можно отнести процессы сравнения уникального серийного номера смарт-карты с номерами, имеющимися в базе данных рабочей станции, сравнение хеш-кода пароля с эталонным хеш-кодом при парольной аутентификации в ОС Windows.

К функции *управления* относятся те элементы алгоритма работы средства аутентификации, которые обеспечивают формирование разрешающего или запрещающего управляющего воздействия. К функции управления, например, можно отнести процесс передачи LUID (разрешающее управляющее воздействие) или статуса отказа (запрещающее управляющее воздействие) в LSASS.

К функции *контроля* относятся те элементы алгоритма работы средства аутентификации, которые обеспечивают проверку соответствия управляющего воздействия, выработанного функцией управления, результатам аутентификации.

Таким образом, блок-схема обобщенного алгоритма работы средства аутентификации должна иметь вид, представленный на рис. 1.11.

Сформулируем ряд утверждений, определяющих полноту и достаточность полученной блок-схемы для представления алгоритма функционирования средства аутентификации.

Утверждение 1.1. Средства аутентификации относятся к классу средств защиты каналов доступа.

Утверждение 1.2. Необходимым и достаточным условием реализации средства аутентификации является наличие в его функциональной структуре совокупности функций обнаружения, опознания, управления и контроля.

Утверждение 1.3. Алгоритм работы средства аутентификации заключается в строгой последовательности выполнения функций обнаружения, опознания, управления и контроля.

Утверждение 1.4. Средство аутентификации должно обеспечивать формирование выходного воздействия только при выполнении полного цикла работы.

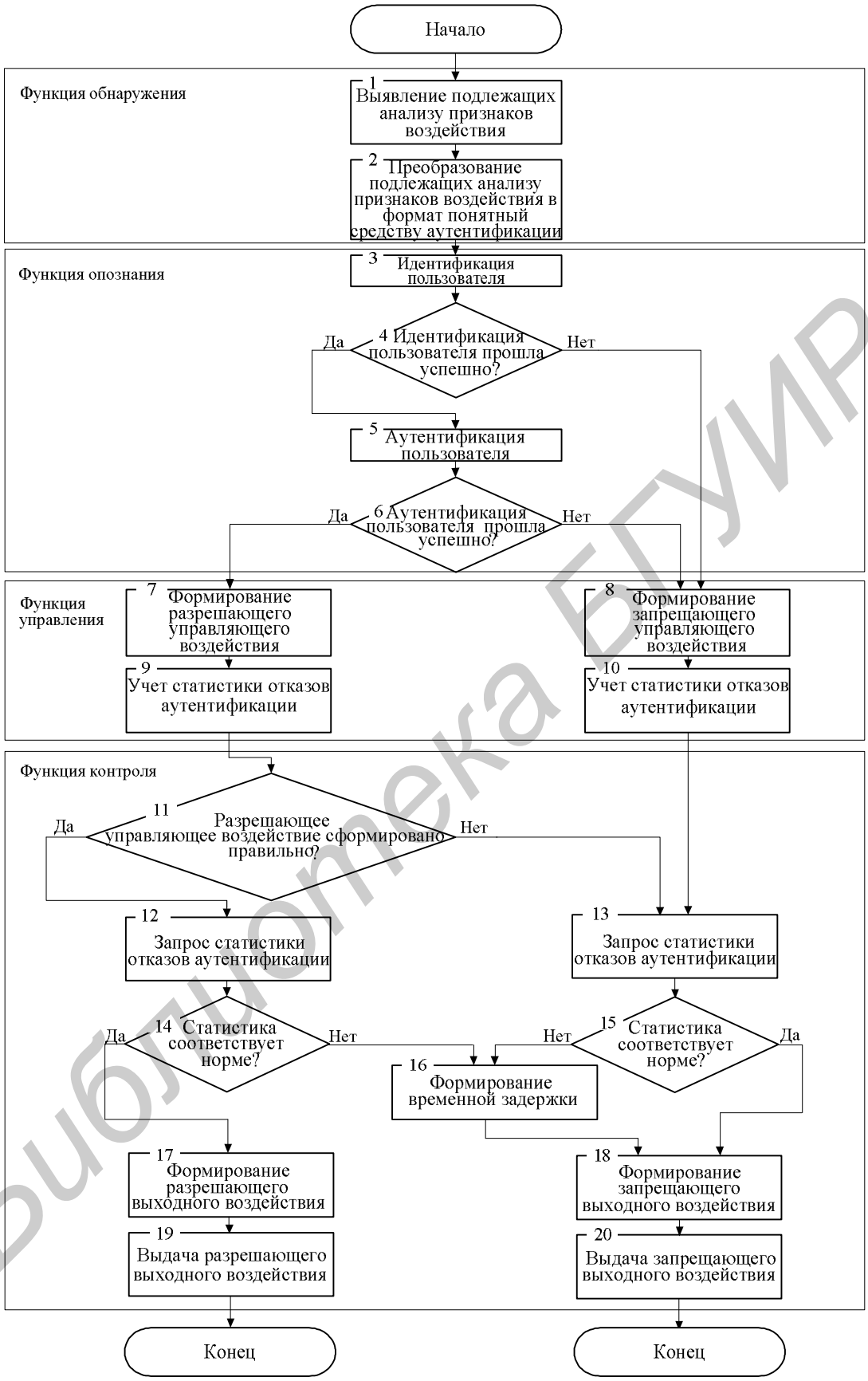

Рис. 1.11. Блок-схема алгоритма работы средства аутентификации

1.5. Эффективность средства аутентификации

Любая техническая система (средство) создается для выполнения определенного набора задач (функций). Выполнение системой (средством) заданного набора задач (функций) назовем *операцией*. Определим эффективность операции как степень соответствия реального (фактического или ожидаемого) результата операции требуемому, или, иными словами, как степень достижения цели операции. Тогда *эффективность* технического средства можно определить как степень выполнения заданного набора функций. Как и всякое свойство, эффективность обладает определенной интенсивностью своего проявления. Меру интенсивности проявления эффективности называют *показателем эффективности E*.

Следовательно, показатель эффективности любой технической системы (средства) есть мера степени соответствия реального достигаемого результата R выполнения операции требуемому результату $R_{\text{ТР}}$.

Для описания соответствия реального результата R операции требуемому $R_{\text{ТР}}$ на множестве результатов операции вводится числовая функция

$$r = r(R(u), R_{\text{ТР}}),$$

где $R(u)$ – реальный результат операции, зависящий от параметров u;

$R_{\text{ТР}}$ – требуемый результат операции;

u – параметры операции, определяемые конструкцией технического средства, набором выполняемых функций, средой функционирования и т.д.

Функция $r(R(u), R_{\text{ТР}})$, называемая функцией соответствия, показывает степень достижения цели операции и в случае детерминированности переменных $R(u)$ и $R_{\text{ТР}}$, может быть использована в качестве показателя эффективности

$$E(u) = r(R(u), R_{\text{ТР}}).$$

Определим вид показателя эффективности для средства аутентификации.

Главной (основной) задачей средства аутентификации является надежное опознание конкретного субъекта. В соответствии с этим показатель эффективности средства аутентификации можно определить как меру приближения вероятности правильного опознания субъекта данным средством в реальных условиях функционирования $P_{\text{ПО}}$ требуемой $P_{\text{ТР}}$. Тогда функцию соответствия ρ а следовательно, и эффективность E средства аутентификации можно определить в виде

$$E = F(P_{\text{ТР}} - P_{\text{ПО}}).$$

Функция F должна обладать определёнными свойствами. При равенстве $P_{\text{ПО}}$ и $P_{\text{ТР}}$ эффективность средства аутентификации является максимальной и должна быть равна единице, а если $P_{\text{ПО}}$ стремится к нулю, то и эффективность снижается и стремится к нулю.

Для определения зависимости эффективности средства аутентификации от меры приближения $P_{ПО}$ к $P_{ТР}$ целесообразно выбрать функцию, удовлетворяющую приведенным выше условиям. Приведенным выше требованиям удовлетворяет математическая функция $E = e^{-2p \cdot \Delta^2}$ (Δ – нормированная мера разности $P_{ТР} - P_{ПО}$), график которой приведен на рис. 1.12:

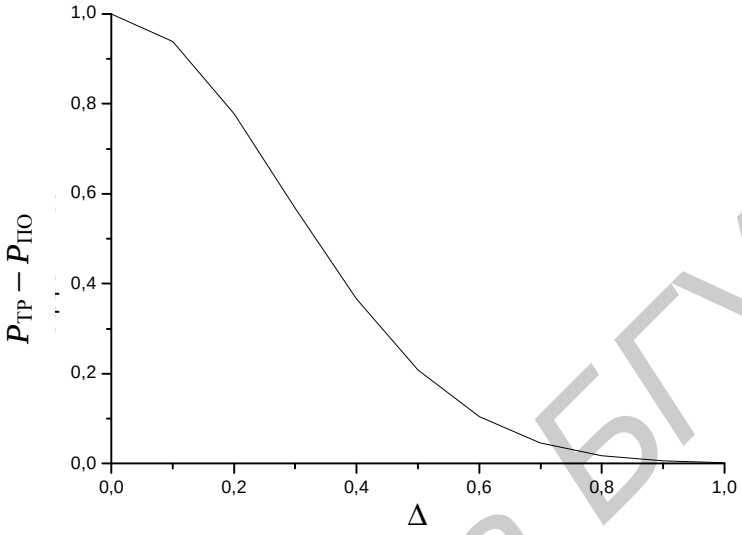

Рис. 1.12. Зависимость эффективности средства аутентификации от меры приближения $P_{ПО}$ к $P_{ТР}$

Нормированная мера приближения $P_{ПО}$ к $P_{ТР}$ определяется следующим выражением:

$$\Delta = 2 \cdot 10^{d+1} \cdot (P_{ТР} - P_{ПО}),$$

где δ – минимальное количество нулевых разрядов после запятой в мерах приближения $P_{ПО}$ к $P_{ТР}$.

Тогда формула для вычисления эффективности средства аутентификации примет вид

$$E = e^{-2p \cdot \left[2 \cdot 10^{d+1} \cdot (P_{ТР} - P_{ПО}) \right]^2}.$$

Вероятность правильного опознания субъекта средством аутентификации в реальных условиях функционирования можно определить как

$$P_{ПО} = 1 - P_{ПЧ},$$

где $P_{ПЧ}$ – вероятность пропуска «чужого» субъекта средством аутентификации.

Средство аутентификации может пропустить «чужого» субъекта в том случае, если произойдет хотя бы одно из следующих событий:

– подбор аутентификатора нарушителем;

– выдача разрешающего выходного сообщения в результате отказа (сбоя) оборудования;

– выдача разрешающего выходного сообщения в результате действий нарушителя.

В таком случае вероятность пропуска «чужого» субъекта $P_{\text{ПЧ}}$ средством аутентификации будет определяться следующим выражением:

$$P_{\text{ПЧ}} = P_{\text{ПА}} + P_{\text{ОТ}} + P_{\text{ДН}} - P_{\text{ОТ}} \cdot P_{\text{ДН}} - P_{\text{ПА}} \cdot P_{\text{ОТ}} - P_{\text{ПА}} \cdot P_{\text{ДН}} + P_{\text{ПА}} \cdot P_{\text{ОТ}} \cdot P_{\text{ДН}},$$

где $P_{\text{ПА}}$ – вероятность подбора аутентификатора;

$P_{\text{ОТ}}$ – вероятность пропуска «чужого» в результате отказов (сбоев) оборудования;

$P_{\text{ДН}}$ – вероятность пропуска «чужого» в результате действий нарушителя.

Отсюда вероятность правильного опознания субъекта средством аутентификации будет иметь вид

$$P_{\text{ПО}} = (1 - P_{\text{ПА}}) \cdot (1 - P_{\text{ОТ}}) \cdot (1 - P_{\text{ДН}}).$$

Тогда формула для вычисления эффективности средства аутентификации будет равна

$$E = F(P_{\text{ТР}} - (1 - P_{\text{ПА}}) \cdot (1 - P_{\text{ОТ}}) \cdot (1 - P_{\text{ДН}})). \qquad (1.1)$$

Рассмотрим способы определения указанных в выражении (1.1) вероятностей.

Вероятность $P_{\text{ПА}}$ зависит от объёма алфавита, длины аутентификатора и является функцией числа попыток подбора

$$P_{\text{ПА}} = 1 - \prod_{i=1}^{k} (1 - P_{\text{П}_i}),$$

где k – число попыток подбора,

$P_{\text{П}_i}$ – вероятность подбора аутентификатора с первой попытки.

Вероятность подбора аутентификатора с первой попытки определяется известной формулой

$$P_{\text{ПА1}} = \frac{1}{A^n},$$

где A – объём алфавита,

n – длина аутентификатора.

Отсюда вероятность подбора аутентификатора с k-й попытки равна

$$P_{\text{ПА}k} = \frac{1}{A^n - k + 1},$$

а подбора за k попыток –

$$P_{\text{ПА1}} = \frac{k}{A^n}.$$

Вероятность $P_{\text{ОТ}}$ определяется надёжностью элементов средства аутентификации и является функцией интенсивности их отказов

$$P_{\text{ОТ}}(l) = 1 - e^{\sum_{j=1}^{n} l_{ij} t},$$

где λ_{ij} – интенсивность отказов элементов, выполняющих i-ю функцию,

n – количество элементов, реализующих i-ю функцию.

Вероятность пропуска «чужого» в результате действия нарушителя $P_{ДН}$ можно определить как произведение вероятностей того, что действие нарушителя было реализовано, и что эта реализация привела к пропуску «чужого»:

$$P_{ДН} = P_{РДН} \cdot P_{ПДН},$$

где $P_{РДН}$ – вероятность того, что действие нарушителя было реализовано,

$P_{ПДН}$ – вероятность того, что реализованное действие нарушителя привело к пропуску «чужого».

В качестве требуемой (расчётной) вероятности правильного опознания выберем вероятность того, что аутентификатор не будет подобран с первой попытки:

$$P_{ТР} = 1 - P_{ПА1}.$$

Вероятность $P_{ПА1}$ (а следовательно, и вероятность $P_{ТР}$) определяется только конструктивными особенностями средства аутентификации, не зависит от внешних и внутренних негативных факторов. Поэтому $P_{ТР}$ может служить верхней границей вероятности $P_{ПО}$.

Таким образом, для оценки эффективности средств аутентификации согласно формуле (1.1), необходимо знать механизмы определения $P_{ОТ}(\lambda)$, не зависящие от класса средства аутентификации, и аналитические выражения для расчёта $P_{ТР}$ применительно к биометрическим средствам.

Контрольные вопросы и задачи

1. Пояснить сущность и особенности классов опознания пользователей в вычислительных сетях.

2. Разработать алгоритм опознания пользователей на основе метода «рукопожатия».

3. Какие требования предъявляются к элементам электронных ключей при их реализации?

5. Почему методы опознания по физическим признакам малопригодны для распределенных систем с большим количеством пользователей?

6. Докажите утверждения 1.1, 1.2, 1.3 и 1.4, определяющие полноту и достоверность функциональной структуры средства аутентификации.

7. Укажите недостатки схемы паролей однократного использования.

8. Дайте определение эффективности технической системы и поясните физический смысл показателя эффективности для средства аутентификации.

9. Если число попыток подбора аутентификатора ограничено числом 10, то как изменится показатель эффективности устройства аутентификации с восьмизначным цифровым паролем?

ГЛАВА 2. ОЦЕНКА ЭФФЕКТИВНОСТИ СРЕДСТВ АУТЕНТИФИКАЦИИ

Для оценки средств аутентификации вне зависимости от их класса рассмотрим способы нахождения $P_{\text{ОТ}}(\lambda)$. Согласно приведенной в предыдущем разделе блок-схеме работы средства аутентификации, алгоритм можно представить в виде вероятностного автомата, содержащего конечный набор состояний, в одном из которых он находится в каждый момент времени. Функционирование вероятностного автомата в каждом такте зависит только от предшествующего состояния.

Графически процесс работы автомата, т.е. переходы из одного состояния в другое, можно представить в виде ориентированного графа. Вершины орграфа определяются состояниями, в которых может находиться автомат. Наличие ребра орграфа определяется наличием перехода между состояниями, которым соответствуют вершины, соединенные данной дугой, причем направление дуги определяет направление перехода из исходного состояния в состояние автомата. Обработка таким автоматом входного воздействия представляет собой последовательный обход ребер от начальной до конечной вершины, причем каждое ребро задействуется только один раз.

Таким образом, автоматное преобразование представляет собой цепь, которая в силу случайности входной последовательности может быть описана марковской цепью.

Полное вероятностное описание процесса функционирования средства аутентификации по теореме умножения вероятностей достигается заданием вероятностей начального состояния и набора вероятностей переходов из одного состояния в другое:

$$P\big(X(t_0), X(t_1)\mathbf{K}, X(t_n)\big) = P_0\big(X(t_0)\big)\prod_{m=1}^{n} P\Big(X(t_m)\Big|X(t_{m-1})\Big), \qquad (2.1)$$

где $X(t_0)$ – начальное состояние средства аутентификации;

$X(t_m)$ – множество состояний, в которых может находиться средство аутентификации в процессе своего функционирования;

$P\Big(X(t_m)\Big|X(t_{m-1})\Big)$ – вероятность перехода от состояния $X(t_{m-1})$ в момент времени t_{m-1} в состояние $X(t_m)$ в момент времени t_m.

Вероятность нахождения средства аутентификации в конкретном состоянии k в момент времени t_n можно представить в виде

$$P_k(t_n) = P\big(X(t_n) = x_k\big). \qquad (2.2)$$

Тогда вероятность перехода средства аутентификации из состояния i в момент времени t_{n-1} в состояние n в момент времени t_n будет иметь вид

$$P_{i,k}(t_{n-1},\, t_n) = P(X(t_{n-1}) = x_i,\; X(t_n) = x_k). \qquad (2.3)$$

Вероятность нахождения средства аутентификации в конкретном состоянии k в момент времени t_n по теореме полной вероятности можно записать в следующем виде:

$$P_k(t_n) = \sum_{i=1}^{K} P_i(t_{n-1}) P_{i,\,k}(t_{n-1},\, t_n)\,. \qquad (2.4)$$

Для полного вероятностного описания процесса функционирования средства аутентификации используется матрица вероятностей одношаговых переходов:

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{1K} \\ P_{21} & P_{22} & P_{2K} \\ \\ P_{K1} & P_{K2} & P_{KK} \end{bmatrix}, \qquad (2.5)$$

где P_{ij} – вероятность перехода средства аутентификации из состояния i (номер строки) в состояние j (номер столбца).

В каждой ячейке матрицы вероятностей одношаговых переходов содержится вероятность перехода из состояния с номером строки в состояние с номером столбца, если данный переход задан на орграфе. Если такой переход не задан на орграфе, то ячейка, соответствующая данному переходу, находится в нулевом состоянии.

Построение орграфа автомата средства аутентификации и его описание с помощью математического аппарата цепей Маркова позволяет определять вероятности пропуска «чужого» или блокировки правомочного субъекта в условиях влияния угроз.

2.1. Построение модели средства аутентификации

Построим модель средства аутентификации, которая описывает средство аутентификации, функционирующее в идеальных условиях, т.е. при отсутствии воздействия внутренних и внешних факторов, приводящих к нарушению выполнения функций. Естественно, подобная модель является упрощённой по сравнению с реальным объектом и находится с ним в отношении сходства, а не тождества. Построение графов модели произведем на основе результатов анализа средств аутентификации, приведенных в гл. 1.

Ниже рассматривается построение модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP путем определения и композиции отдельных графов, соответствующих четырем функциям средства аутентификации: обнаружения, опознания, управления и контроля.

Граф модели функции обнаружения средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP представлен на рис. 2.1.

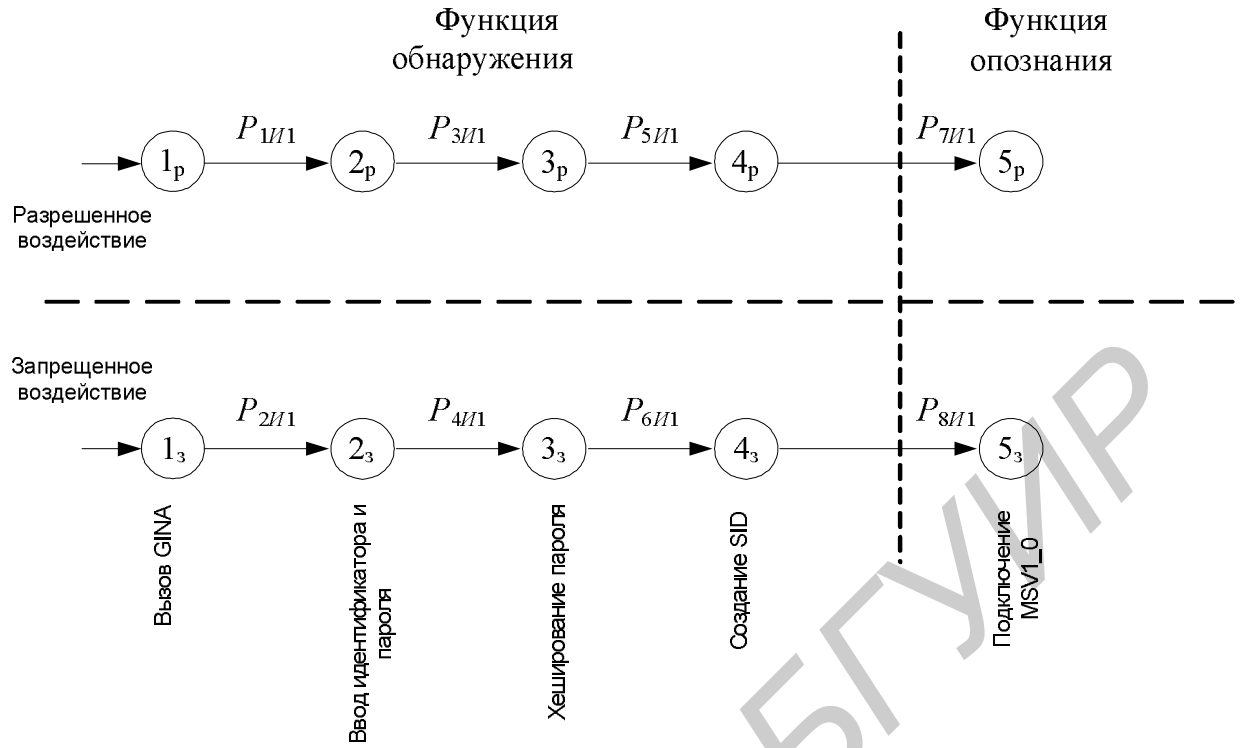

Рис. 2.1. Граф функции обнаружения модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

На графе (рис. 2.1) введены следующие обозначения состояний и вероятностей

$1_р$, $1_з$ – вызов GINA;

$2_р$, $2_з$ – ввод пользователем идентификатора и пароля;

$3_р$, $3_з$ – хеширования пароля;

$4_р$, $4_з$ – создание SID;

$P_{1И1}$ – вероятность того, что при разрешенном воздействии будет произведен вывод полей для ввода пользователем своего идентификатора и пароля и что пользователь их введет;

$P_{2И1}$ – вероятность того, что при запрещенном воздействии будет произведен вывод полей для ввода пользователем своего идентификатора и пароля и что пользователь их введет;

$P_{3И1}$ – вероятность того, что при разрешенном воздействии будет выработан правильный хеш-код;

$P_{4И1}$ – вероятность того, что при запрещенном воздействии будет выработан неправильный хеш-код;

$P_{5И1}$ – вероятность того, что при разрешенном воздействии Winlogon создаст SID;

$P_{6И1}$ – вероятность того, что при запрещенном воздействии Winlogon создаст SID.

Граф модели функции опознания средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP представлен на рис. 2.2.

На графе введены следующие обозначения состояний и вероятностей:

$5_р$, $5_з$ – подключение MSV1_0;

$6_р$, $6b_з$ – положительный результат поиска в базе данных учетной записи с идентификатором пользователя;

$6a_з$ – отрицательный результат поиска в базе данных учетной записи с идентификатором пользователя;

$7_р$ – положительный результат сравнения хешированных паролей;

$7_з$ – отрицательный результат сравнения хешированных паролей;

$8_р$ – генерация LUID при положительном результате сравнения хешированных паролей;

$8_з$ – отсутствие процесса генерации LUID при отрицательном результате сравнения хешированных паролей;

$P_{7И1}$ – вероятность того, что при разрешенном воздействии Winlogon произведет вызов LSASS и LSASS подключит MSV1_0;

$P_{8И1}$ – вероятность того, что при запрещенном воздействии Winlogon произведет вызов LSASS и LSASS подключит MSV1_0;

$P_{9И1}$ – вероятность того, что при разрешенном воздействии поиск в базе данных учетной записи с идентификатором пользователя даст положительный результат;

$P_{10И1}$ – вероятность того, что при запрещенном воздействии поиск в базе данных учетной записи с идентификатором пользователя даст положительный результат;

$P_{11И1}$ – вероятность того, что при запрещенном воздействии поиск в базе данных учетной записи с идентификатором пользователя даст отрицательный результат;

$P_{12И1}$ – вероятность того, что при разрешенном воздействии сравнение хешированных паролей даст положительный результат;

$P_{13И1}$ – вероятность того, что при запрещенном воздействии и положительном результате поиска в базе данных учетной записи с идентификатором пользователя сравнение хешированных паролей даст отрицательный результат;

$P_{14И1}$ – вероятность того, что при отрицательном результате поиска в базе данных учетной записи с идентификатором пользователя будет сформирован отрицательный результат опознания;

$P_{15И1}$ – вероятность того, что при положительном результате сравнения хешированных паролей будет сформирован положительный результат опознания;

$P_{16И1}$ – вероятность того, что при отрицательном результате сравнения хешированных паролей будет сформирован отрицательный результат опознания.

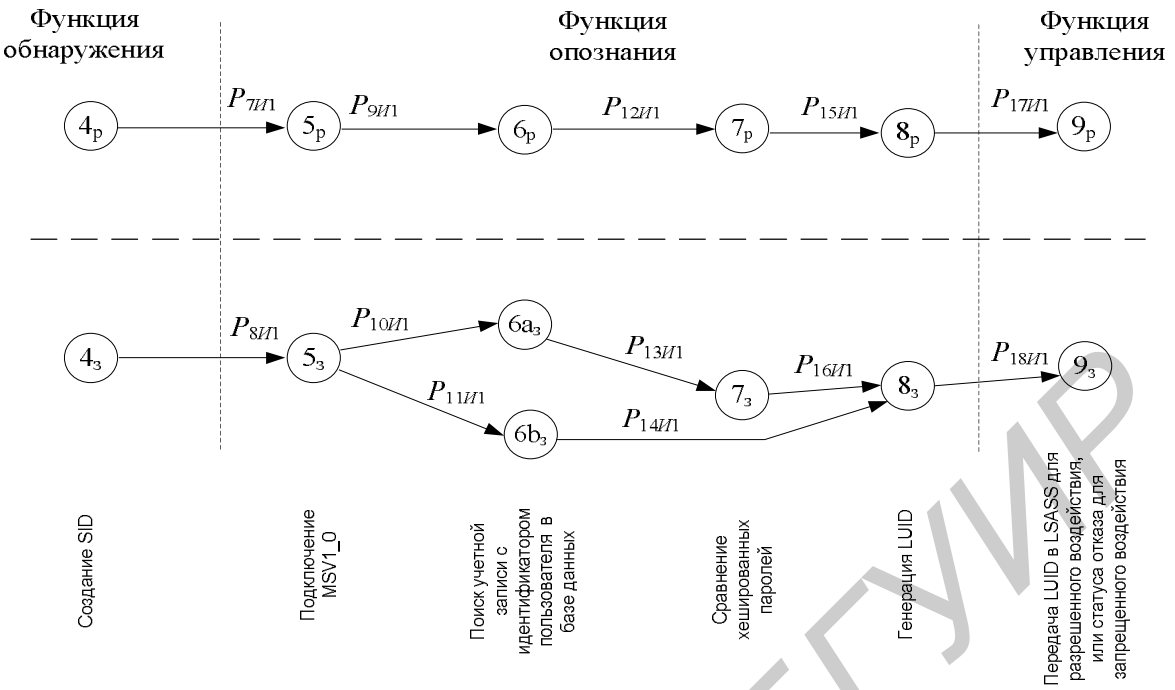

Рис. 2.2. Граф функции опознания модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

Граф модели функции управления средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP представлен на рис. 2.3.

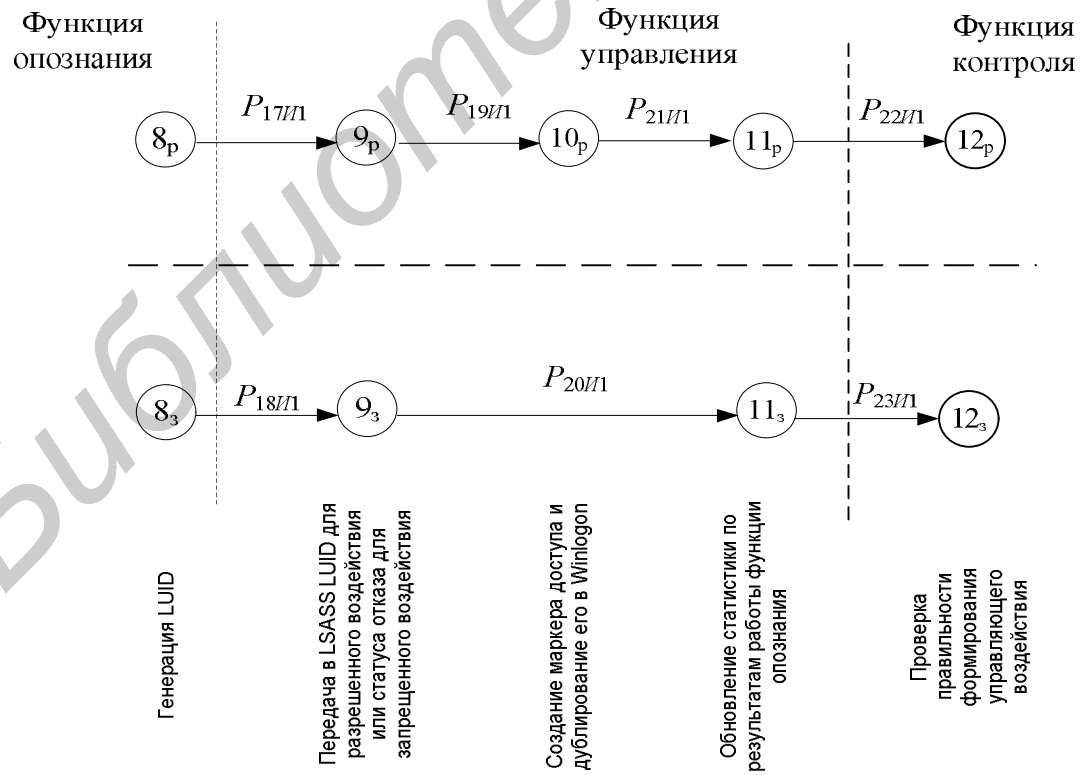

Рис. 2.3. Граф функции управления модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

На графе (рис. 2.3) введены следующие обозначения состояний и вероятностей:

$9_р$ – передача LUID в LSASS;

$9_з$ – передача статуса отказа в LSASS;

$10_р$ – создание маркера доступа и дублирование его в Winlogon;

$11_р$, – обновление статистики положительным результатом аутентификации;

$11_з$ – обновление статистики отрицательным результатом аутентификации;

$P_{17И1}$ – вероятность того, что после формирования положительного результата опознания будет выработано разрешающее управляющее воздействие;

$P_{18И1}$ – вероятность того, что после формирования отрицательного результата опознания будет выработано запрещающее управляющее воздействие;

$P_{19И1}$ – вероятность того, что при разрешенном воздействии будет создан маркер доступа;

$P_{20И1}$ – вероятность того, что при запрещенном воздействии будет произведено обновление статистики отрицательным результатом опознания;

$P_{21И1}$ – вероятность того, что после создания маркера доступа будет произведено обновление статистики положительным результатом опознания.

Граф модели функции контроля средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP представлен на рис. 2.4.

На графе (рис. 2.4) введены следующие обозначения состояний и вероятностей:

$12_р$, $12_з$ – проверка правильности формирования управляющего воздействия;

$13_з$ – обработка статистики количества неправильных попыток ввода идентификатора и пароля;

$14_р$ – генерация и выдача на выход средства аутентификации разрешающего выходного воздействия;

$14a_з$ – блокировка средства аутентификации;

$14b_з$, – генерация и выдача на выход средства аутентификации запрещающего выходного воздействия;

$P_{22И1}$ – вероятность того, что при разрешенном воздействии после обновления статистики положительным результатом опознания проверка правильности формирования управляющего воздействия оставит средство аутентификации в разрешающем состоянии;

$P_{23И1}$ – вероятность того, что при запрещенном воздействии после обновления статистики отрицательным результатом опознания проверка правильности формирования управляющего воздействия оставит средство аутентификации в запрещающем состоянии;

$P_{24И1}$ – вероятность того, что при разрешенном воздействии будет сгенерировано и выдано на выход средства аутентификации разрешающее выходное воздействие;

$P_{25И1}$ – вероятность того, что при запрещенном воздействии будет проведена обработка статистики количества неправильных попыток ввода идентификатора и пароля;

$P_{26И1}$ – вероятность того, что при запрещенном воздействии в случае превышения числа разрешенных попыток ввода идентификатора и пароля средство аутентификации будет заблокировано;

$P_{27И1}$ – вероятность того, что при запрещенном воздействии в случае отсутствия превышения числа разрешенных попыток ввода идентификатора и пароля будет сгенерировано и выдано на выход средства аутентификации запрещающее выходное воздействие.

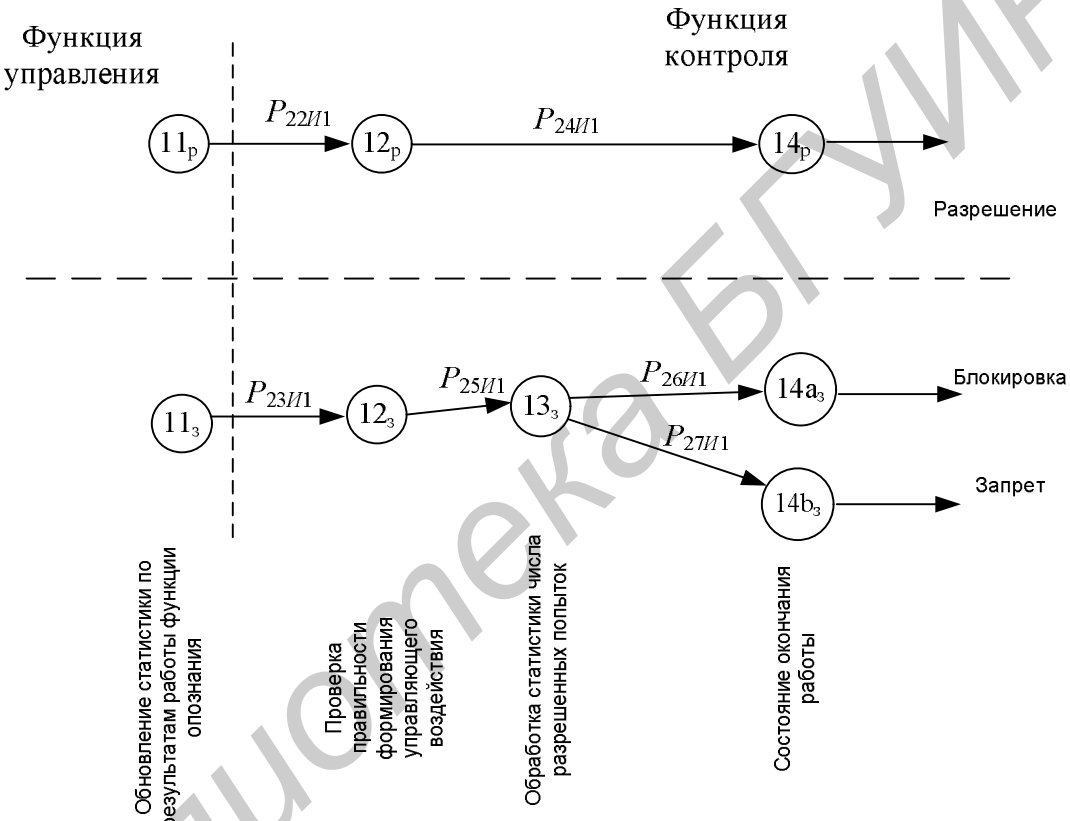

Рис. 2.4. Граф функции контроля модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

На рис. 2.5 представлен полный граф модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP при интерактивном входе. В данном графе при наличии разрешенного входного воздействия, на выходе всегда будет сгенерировано разрешающее выходное воздействие и наоборот, при наличии запрещенного входного воздействия, на выходе всегда будет сгенерировано запрещающее выходное воздействие.

Аналогичным образом могут быть построены графы моделей других рассмотренных в гл. 1 средств аутентификации.

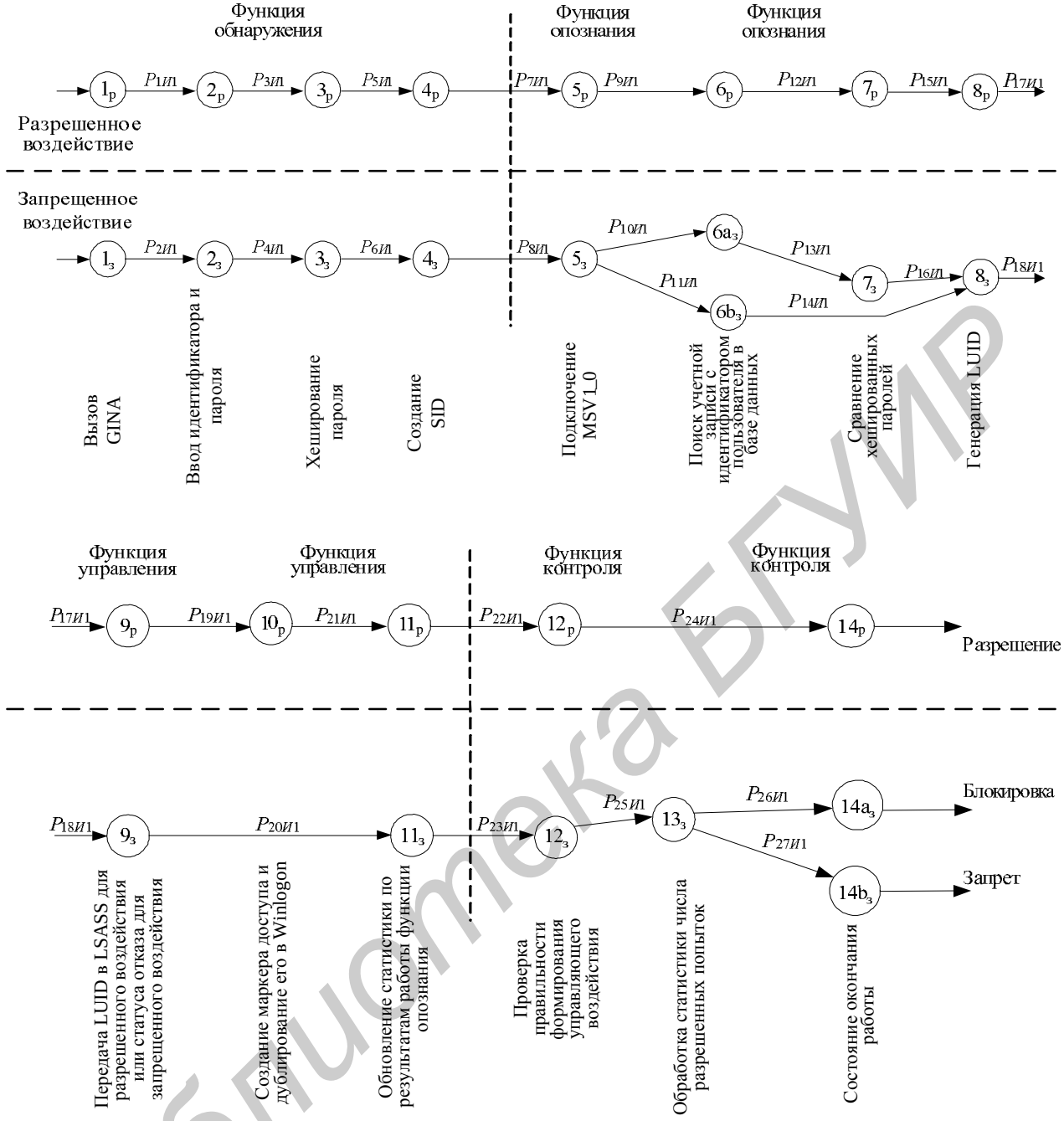

Рис. 2.5. Граф модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

2.2. Модель средства аутентификации с учетом действия угроз

Моделью средства аутентификации, учитывающей действия угроз, назовем модель, которая описывает средство аутентификации в условиях воздействия внутренних и внешних факторов, приводящих к нарушению выполнения его функций. Данные факторы отображаются на графе модели новыми состояниями и враждебными переходами, которые соответствуют конкретным угрозам, рассмотренным в предыдущем разделе.

Под *угрозами безопасности* средству аутентификации определим возможные события, приводящие к тому, что средство аутентификации может пропустить «чужого» или блокировать правомочного субъекта.

Источники угроз безопасности средств аутентификации можно разделить на две группы:

- преднамеренные;
- случайные.

Источником преднамеренных угроз, которые необходимо учесть при моделировании средства аутентификации, являются несанкционированные действия пользователей системы.

Источниками случайных угроз могут быть:

- отказы (сбои) в работе оборудования средства аутентификации;
- ошибки проектирования средства аутентификации;
- ошибки эксплуатации средства аутентификации.

В качестве угроз будем рассматривать события, приводящие к переходу средства аутентификации из запрещающего состояния в разрешающее при запрещенном входном воздействии и из разрешающего состояния в запрещающее при разрешенном входном воздействии.

В табл. 2.1 приведен перечень ситуаций, приводящих к реализации угроз средству аутентификации с использованием паролей в ОС MS Windows XP.

Виды угроз безопасности Таблица 2.1

1. Преднамеренные
1.1. Несанкционированные действия пользователей системы
1.1.1. Очистка статистики количества неверных попыток входа в систему
1.1.2. Изменение статистики количества неверных попыток входа в систему
1.1.3. Изменение алгоритма хеширования пароля, при котором запрещенный пароль будет преобразован в разрешенный хеш-код
2. Случайные
2.1. Ошибки проектирования и эксплуатации средства аутентификации
2.1.1. Посимвольная проверка пароля
2.1.2. Возврат в начальное состояние после отрицательного результата идентификации
2.1.3. Возврат в начальное состояние после отрицательного результата аутентификации
2.1.4. Возврат в начальное состояние после генерации отрицательного результата работы функции опознания
2.1.5. Возврат в начальное состояние после генерации запрещенного управляющего воздействия
2.1.6. Переход в конечное состояние, минуя функции управления и контроля, после отрицательного результата работы функции опознания
2.1.7. Переход в конечное состояние, минуя функцию контроля, после отрицательного результата работы функции управления
2.2. Сбои в работе оборудования
2.2.1. Неверное хеширование пароля, при котором запрещенный пароль будет преобразован в разрешенный хеш-код
2.2.2. Неверное хеширование пароля, при котором разрешенный пароль будет преобразован в запрещенный хеш-код

Окончание табл. 2.1

2.2.3. Положительный результат поиска в базе данных учетной записи с несуществующим идентификатором
2.2.4. Отрицательный результат поиска в базе данных учетной записи с существующим идентификатором
2.2.5. Неверное сравнение хешированных паролей, при котором идентичные пароли будут признаны неидентичными
2.2.6. Неверное сравнение хешированных паролей, при котором неидентичные пароли будут признаны идентичными
2.2.7. Генерация неправильного LUID
2.2.8. Передача пустого LUID в LSASS вместо статуса отказа
2.2.9. Передача статуса отказа в LSASS вместо LUID
2.2.10. Отказ в создании маркера доступа
2.2.11. Создание ложного маркера доступа
2.2.12. Ошибочное обновление статистики отрицательным результатом работы функции опознания
2.2.13. Выработка неправильного результата проверки правильности формирования разрешающего управляющего воздействия
2.2.14. Выработка неправильного результата проверки правильности формирования запрещающего управляющего воздействия
2.2.15. Переход в запрещающее состояние в процессе работы функции контроля
2.2.16. Переход в разрешающее состояние в процессе работы функции контроля
2.2.17. Случайная очистка статистики количества неверных попыток входа в систему
2.2.18. Случайное установление превышения статистики количества неверных попыток входа в систему

Граф модели рассмотренного выше средства аутентификации с учётом действия угроз представлен на рис. 2.6. Рассмотрим враждебные переходы по каждой функции отдельно и дадим определения обозначенным на графе (рис. 2.6) вероятностям:

а) *изменения в работе функции обнаружения:*

Q_{1P1} – вероятность того, что при разрешенном воздействии будет выработан неправильный хеш-код;

Q_{2P1} – вероятность того, что при запрещенном воздействии будет выработан правильный хеш-код;

б) *изменения в работе функции опознания:*

Q_{3P1} – вероятность того, что при разрешенном воздействии поиск в базе данных учетной записи с идентификатором пользователя даст отрицательный результат;

Q_{4P1} – вероятность того, что при разрешенном воздействии сравнение хешированных паролей даст отрицательный результат;

Q_{5P1} – вероятность того, что при запрещенном воздействии сравнение хешированных паролей даст положительный результат;

Q_{6P1} – вероятность того, что после отрицательного результата поиска в базе данных учетной записи с идентификатором пользователя средство аутентификации вернется в состояние ожидания входного воздействия;

Q_{7P1} – вероятность того, что при положительном результате сравнения хешированных паролей будет сформирован отрицательный результат опознания;

Q_{8P1} – вероятность того, что при отрицательном результате сравнения хешированных паролей будет сформирован положительный результат опознания;

Q_{9P1} – вероятность того, что после формирования отрицательного результата сравнения хешированных паролей средство аутентификации вернется в состояние ожидания входного воздействия;

Q_{12P1} – вероятность того, что после формирования отрицательного результата опознания средство аутентификации вернется в состояние ожидания входного воздействия;

в) *изменения в работе функции управления:*

Q_{10P1} – вероятность того, что после формирования положительного результата опознания будет выработано запрещающее управляющее воздействие;

Q_{11P1} – вероятность того, что после формирования отрицательного результата опознания будет выработано разрешающее управляющее воздействие;

Q_{13P1} – вероятность того, что при разрешенном воздействии будет произведено обновление статистики отрицательным результатом опознания;

Q_{14P1} – вероятность того, что при запрещенном воздействии будет создан маркер доступа;

Q_{15P1} – вероятность того, что после создания маркера доступа будет произведено обновление статистики отрицательным результатом опознания;

г) *изменения в работе функции контроля:*

Q_{16P1} – вероятность того, что при разрешенном воздействии после обновления статистики положительным результатом опознания проверка правильности формирования управляющего воздействия переведет средство аутентификации в запрещающее состояние;

Q_{17P1} – вероятность того, что при запрещенном воздействии после обновления статистики отрицательным результатом опознания проверка правильности формирования управляющего воздействия переведет средство аутентификации в разрешающее состояние;

Q_{18P1} – вероятность того, что при разрешенном воздействии после проверки правильности формирования управляющего воздействия будет произведена обработка статистики числа разрешенных попыток ввода идентификатора и пароля;

Q_{19P1} – вероятность того, что при запрещенном воздействии после проверки правильности формирования управляющего воздействия будет сгенерировано и выдано на выход средства аутентификации разрешающее выходное воздействие;

Q_{20P1} – вероятность того, что после обработки статистики числа разрешенных попыток ввода идентификатора и пароля будет сгенерировано и выдано на выход средства аутентификации разрешающее выходное воздействие.

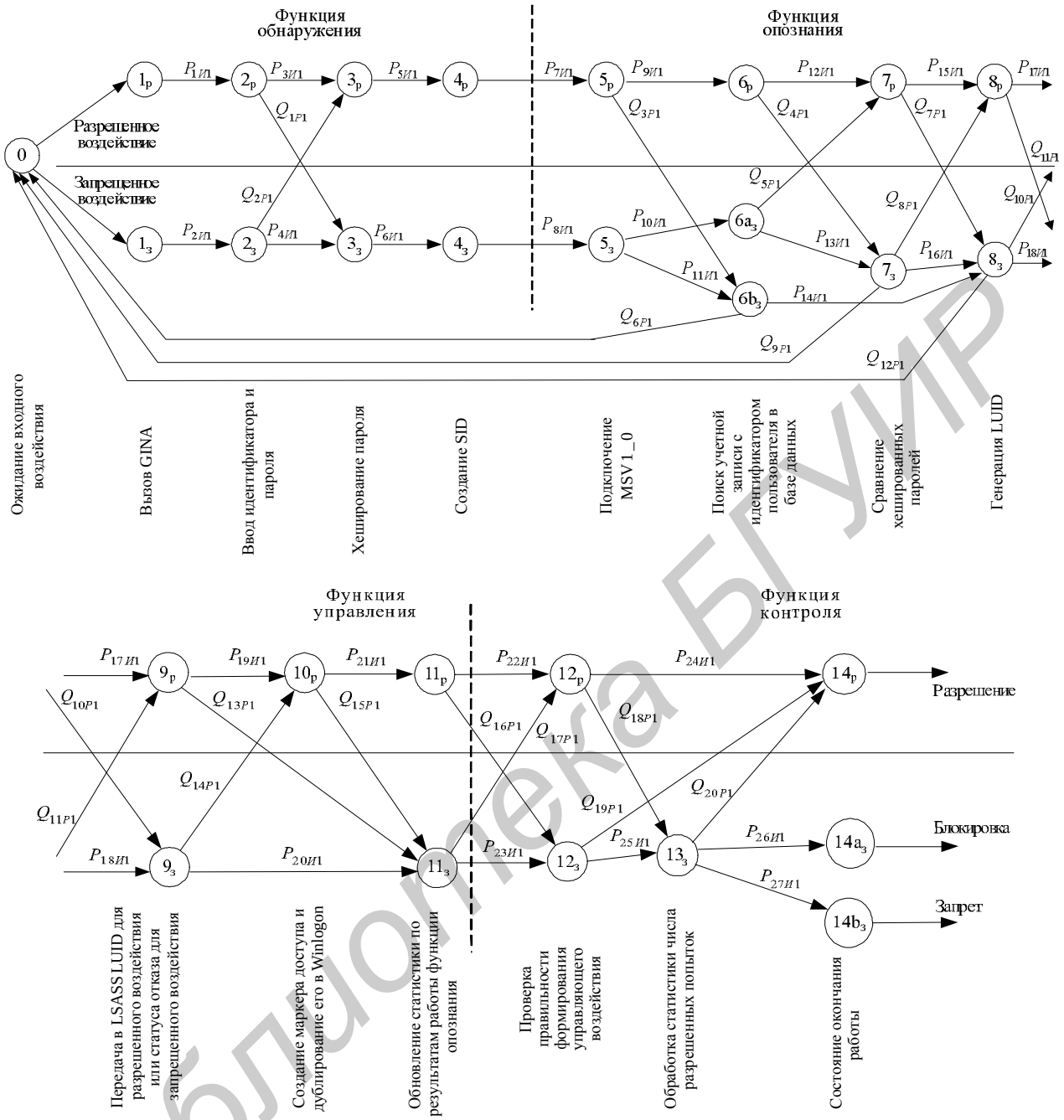

Рис. 2.6. Граф модели средства аутентификации с использованием паролей в операционной системе Microsoft Windows XP

2.3. Реализация модели средств аутентификации

Модель средств аутентификации реализована в виде программы и используется для нахождения вероятности правильного опознания в условиях появления неисправностей и воздействия нарушителя. Она состоит из четырех отдельных информационных областей:

1. *Исходные данные*. В этой информационной области происходит ввод исходных данных для работы программы. Исходные данные включают в себя:

– тип средства аутентификации;

– вид входного воздействия;

– значения вероятностей враждебных переходов.

2. *Граф средства аутентификации*. В данной информационной области осуществляется построение и отображение графа заданного средства аутентификации согласно исходным данным.

3. *Матрица вероятностей переходов*. В этой области формируется и отображается матрица переходов, соответствующая графу средства аутентификации.

4. *Результаты работы программы*. В этой информационной области представлены результаты работы программы. Значения вероятности правильного опознания субъекта средством аутентификации представлены в виде графика. Кроме того, для данного средства аутентификации рассчитываются значения вероятностей пропуска «чужого» субъекта, блокировки средства аутентификации, возврата в начальное состояние.

Основное окно программы с открытой информационной областью для ввода исходных данных представлено на рис. 2.7.

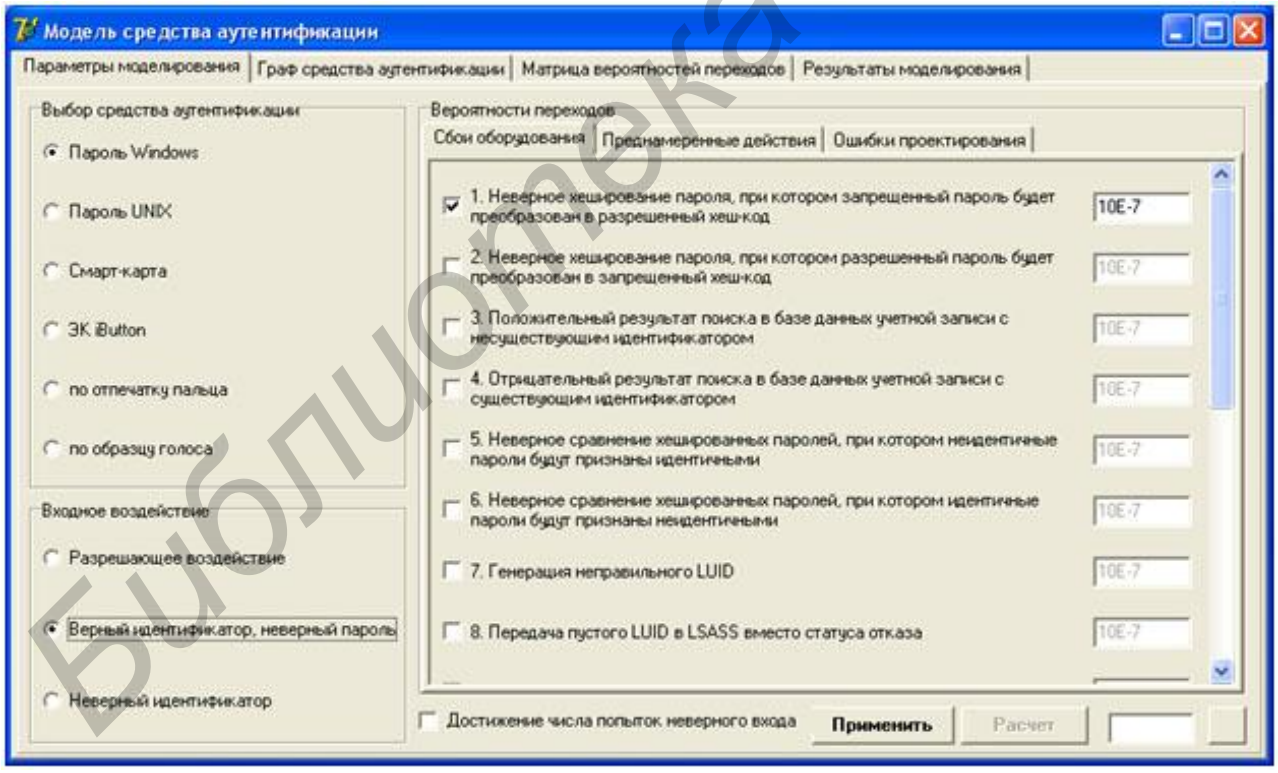

Рис. 2.7. Информационная область программы для ввода исходных данных

В первую очередь выбирается одно из шести предложенных программой средств аутентификации (по умолчанию – пароль Windows). Далее выбирается

одно из возможных в данном средстве аутентификации входных воздействий (по умолчанию разрешающее). После этого помечаются угрозы, воздействующие на средство аутентификации, напротив их названий в трех закладках: «Неисправности оборудования», «Преднамеренные действия» и «Ошибки проектирования». Здесь же задаются вероятности для неисправностей оборудования. Преднамеренные действия нарушителя и ошибки проектирования вносятся в модель как реализованные события с вероятностью, равной 1. В последнюю очередь устанавливается условие ограничения попыток неверного входа в систему. Наличие метки напротив надписи «Достижение числа попыток неверного входа» свидетельствует об учете данного события в модели.

После нажатия кнопки «Применить» по заданным параметрам строится граф средства аутентификации и матрица переходов. Отобразить граф или матрицу переходов можно, нажав на соответствующие закладки вверху окна программы. Примеры построения графа средства аутентификации по паролю в ОС MS Windows XP и его матрицы переходов представлены соответственно на рис. 2.8. и 2.9. Далее производится определение всех возможных вариантов путей прохода графа от заданного входного воздействия к разрешающему или запрещающему выходным воздействиям, к начальному состоянию или к состоянию блокировки средства аутентификации, и осуществляется расчёт соответствующих вероятностей.

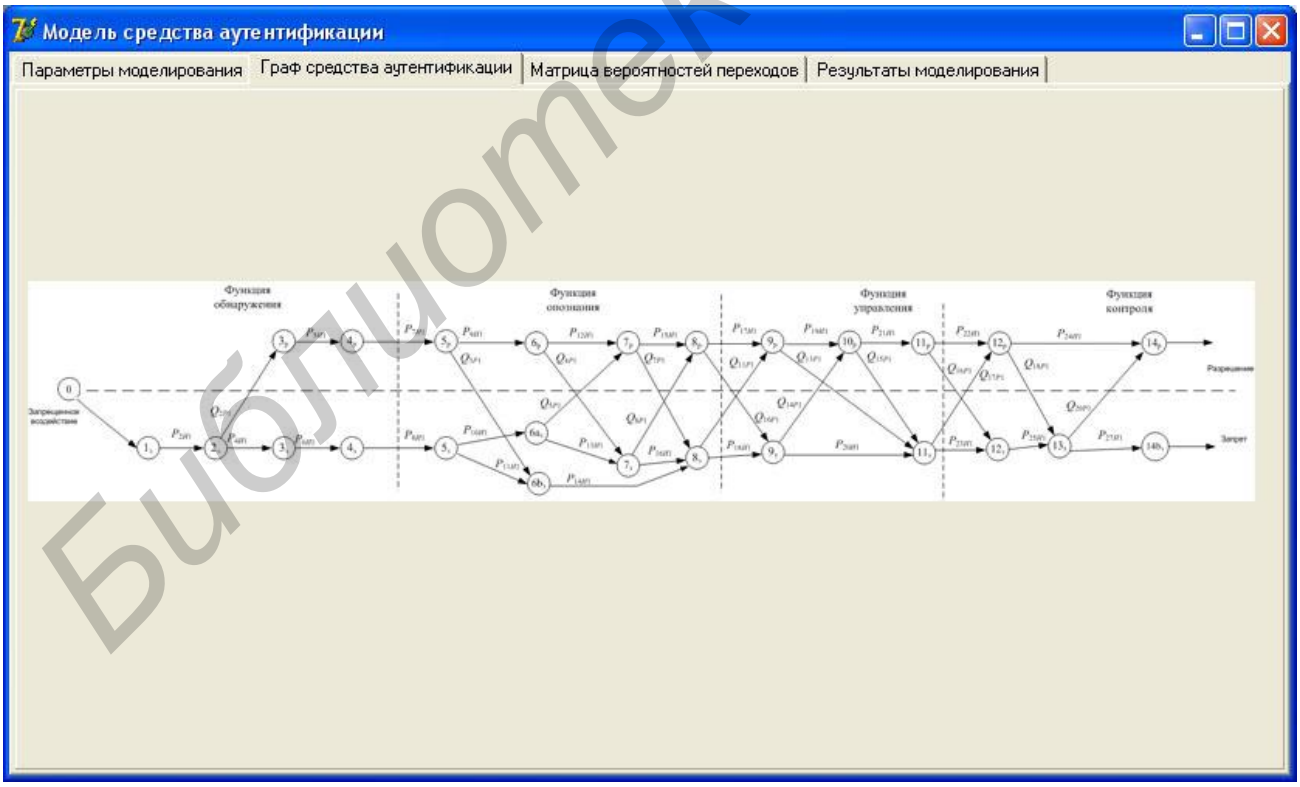

Рис. 2.8. Информационная область программы с графом средства аутентификации

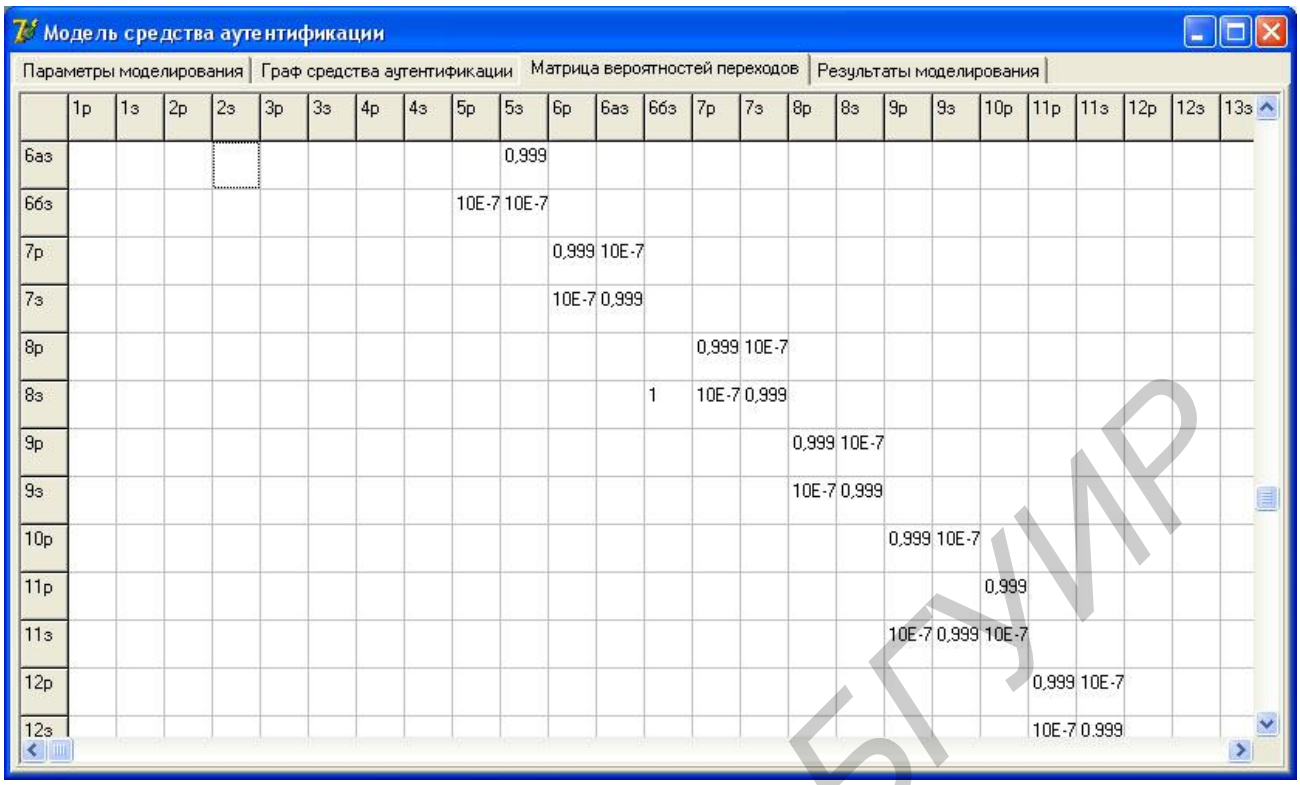

Рис. 2.9. Информационная область программы с матрицей переходов

На рис. 2.10 представлен график зависимости вероятности правильного опознания от интенсивности отказов для средства аутентификации по паролю в ОС MS Windows XP.

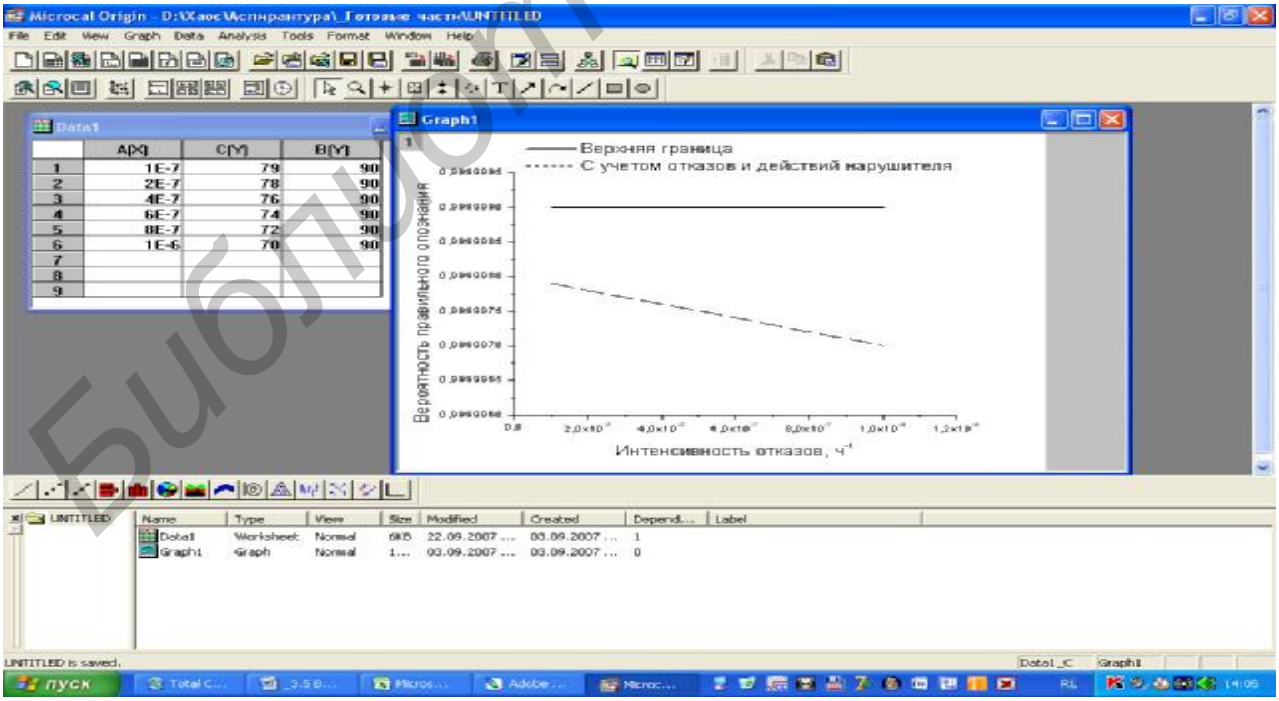

Рис. 2.10. График зависимости вероятности правильного опознания
от интенсивности отказов для средства аутентификации
по паролю в ОС MS Windows XP

2.4. Определение вероятности пропуска «чужого» субъекта средством аутентификации по отпечатку пальца

Вероятность пропуска «чужого» субъекта данным средством аутентификации определяется как вероятность совпадения изображения эталонного отпечатка пальца с изображением отпечатка пальца, предоставленного субъектом.

Основным подходом к установлению совпадения отпечатков пальцев является поиск и сопоставление особенностей рисунков этих отпечатков, на которых представлены «окончания» и «раздвоения» папиллярных линий. Данные особенности отпечатка пальца носят название *минуций*, а их совокупность – *вектора минуций*.

Совпадение изображений отпечатков пальцев устанавливается по совпадению векторов минуций, выделенных на эталонном и предоставленном субъектом изображениях, причем каждая минуция в векторе минуций представлена двумя координатами и углом ориентации.

В процессе сравнения изображение отпечатка пальца разбивается на конечное число секторов, каждый из которых представляет собой квадрат размером $n \cdot n$ пикселов.

После бинаризации и утончения изображения отпечатка пальца каждый пиксел этого изображения может принимать всего два цвета: черный или белый. Следовательно, существует конечное число вариантов изображений, которые могут быть реализованы в секторе размером $n \times n$ пикселов. Количество вариантов определяется по формуле

$$S = 2^{(n \times n)}, \tag{2.6}$$

где S – количество вариантов изображений, которые могут быть реализованы в секторе размером $n \times n$ пикселов;

n – количество пикселов, составляющее сторону квадратного сектора.

Вероятность того, что любая из минуций, выделенных на изображении отпечатка пальца (p), предоставленного субъектом, совпадет с любой из минуций (без учета ее типа), выделенных на эталонном изображении отпечатка пальца (q) (или, другими словами, минуция, выделенная на предоставленном субъектом изображении отпечатка пальца, будет иметь координаты и ориентацию, соответствующие координатам и ориентации минуции эталонного изображения отпечатка пальца), можно свести к вероятности появления в нужном секторе предоставляемого субъектом изображения отпечатка пальца набора пикселов, соответствующего изображению минуции в этом же секторе эталонного изображения отпечатка пальца. Данная вероятность определяется по формуле

$$P_{\text{мин}} = \frac{Z}{S}, \tag{2.7}$$

где $P_{\text{мин}}$ – вероятность совпадения одной выделенной на предоставленном субъектом изображении отпечатка пальца минуции с эталонной;

Z – количество вариантов реализации минуции без учета ее типа в секторе размером $n \times n$ пикселов.

Таким образом, вероятность того, что все p выделенные на изображении отпечатка пальца субъекта минуции совпадут с эталонными минуциями, можно свести к вероятности появления в нужных p секторах изображения отпечатка пальца субъекта наборов пикселов, соответствующих изображению минуции в этих же секторах эталонного изображения отпечатка пальца. Вероятность совпадения всех p выделенных на предоставленном субъектом изображении отпечатка пальца минуций с эталонными вычисляется по формуле

$$P(p) = (\frac{Z}{S})^p, \tag{2.8}$$

где $P(p)$ – вероятность совпадения p выделенных на предоставленном субъектом изображении отпечатка пальца минуций с эталонными.

В реальных условиях количество минуций, выделенных на изображении отпечатка пальца субъекта (p), и количество минуций, выделенных на эталонном изображении отпечатка пальца (q), различно. Поэтому для установления совпадения изображений отпечатков пальцев используют порог меры близости, который определяется выражением

$$g = \frac{D^2}{p \cdot q}, \tag{2.9}$$

где g – порог меры близости изображений отпечатков пальцев;

D – количество совпавших пар минуций.

Если при заданном количестве минуций, выделенных на изображении отпечатка пальца субъекта (p), минуций, выделенных на эталонном изображении отпечатка пальца (q) и полученном количестве совпавших минуций порог меры близости будет меньше заданного, то предоставленное субъектом изображение отпечатка пальца считается несовпавшим с эталонным и субъект, предоставивший его, считается «чужим». Если в подобной ситуации порог меры близости будет больше заданного, то предоставленное субъектом изображение отпечатка пальца считается совпавшим с эталонным.

Задав порог меры близости изображений отпечатков пальцев, выше которого изображения считаются совпавшими, можно определить критическую область количества совпавших пар минуций (D). Используя формулу (2.9) и зная, что количество совпавших пар минуций – это целая величина, получим критическую область количества совпавших пар минуций для заданного числа минуций, выделенных на изображении отпечатка пальца субъекта, определяемую неравенством

$$\text{int}(\sqrt{g \cdot p \cdot q}) + 1 \le D \le \min(p, q). \tag{2.10}$$

Если количество совпавших пар минуций будет находиться в этой области, то изображения отпечатков пальцев будут считаться одинаковыми.

Следовательно, два изображения отпечатков пальцев считаются неодинаковыми, если количество совпавших пар минуций не будет находиться в данной области или если данная область является пустой и выполняется следующее неравенство:

$$\min(p, q) < \text{int}(\sqrt{g \cdot p \cdot q}) + 1. \tag{2.11}$$

Таким образом, формула вероятности пропуска «чужого» субъекта данным средством аутентификации для заданного числа минуций, выделенных на изображении отпечатка пальца, предоставленного субъектом, имеет вид

$$P_{\text{ПГ}} = \sum_{i=\text{int}(\sqrt{g \cdot p \cdot q})+1}^{\min(p,q)} P(p_i) = \sum_{i=\text{int}(\sqrt{g \cdot p \cdot q})+1}^{\min(p,q)} \left(\frac{Z}{S}\right)^i, \qquad (2.12)$$

где i – количество совпавших пар минуций.

В формуле (2.12) значения количества вариантов изображений S и количества вариантов реализации минуции без учета ее типа Z, которые могут быть реализованы в секторе размером $n \times n$ пикселов зависят от размера сектора.

После бинаризации и утончения изображения отпечатка пальца с учетом ряда приближений, минуцию, независимо от ее типа и угла ориентации, можно выделить на секторе с минимальным размером 3×3 пиксела, причем координаты данного сектора и будут координатами выделенной минуции.

Минуция типа «окончание» папиллярной линии возникает в том случае, когда центральный пиксел области закрашен, а среди соседних восьми пикселов закрашены один или два смежных пиксела. В секторе размером 3×3 пиксела существует 16 вариантов реализации минуции типа «окончание» папиллярной линии.

Минуция «раздвоение» папиллярной линии возникает в том случае, когда центральный пиксел области закрашен, а среди соседних восьми пикселов закрашены три отдельные области пикселов, которые могут быть представлены одиночным или двумя смежными пикселами. В секторе размером 3·3 пиксела существует 48 вариантов реализации минуции типа «раздвоение» папиллярной линии.

Сектор размером 3×3 пиксела включает в себя девять пикселов, цвет которых может принимать лишь два значения. Следовательно, существует всего $2^9 = 512$ вариантов бинарных изображений в секторе размером 3×3 пиксела. С другой стороны, в секторе размером 3×3 пиксела существует всего 16 + 48 = 64 варианта реализации минуции. Тогда вероятность совпадения одной выделенной на предоставленном субъектом изображении отпечатка пальца минуций с эталонной равна

$$P_{\text{мин}} = \frac{64}{2^{(3 \cdot 3)}} = \frac{64}{512} = \frac{1}{8}.$$

Кроме того, после бинаризации и утончения изображения отпечатка пальца папиллярные линии на изображении встречаются реже, чем области между ними. Для пиксельного представления изображения это сводится к тому, что белые пикселы встречаются чаще, чем черные. Поэтому можно утверждать, что вероятность появления в секторе размером 3×3 пиксела изображения с двумя и менее белыми пикселами ничтожно мала. Количество изображений, содержащих в секторе два, один или нуль белых пиксела определяется по формуле

$$N = \sum_{i=0}^{3} \frac{H!}{i! \cdot (H-i)!}, \qquad (2.13)$$

где H – общее количество пикселов в секторе.

Подставив в формулу (2.31) количественные значения, получим

$$N = \sum_{i=0}^{3} \frac{9!}{i!\cdot(9-i)!} = 1 + 9 + 36 = 46 .$$

На основании этого формулу вероятности пропуска «чужого» субъекта можно упростить:

$$P_{\text{ПГ}} = \sum_{i=\text{int}(\sqrt{g\cdot p\cdot q})+1}^{\min(p,q)} \left(\frac{64}{512-46}\right)^{i} = \sum_{i=\text{int}(\sqrt{g\cdot p\cdot q})+1}^{\min(p,q)} (0{,}1373)^{i} . \qquad (2.14)$$

Разбиение изображения отпечатка пальца на сектора размером 5×5 пикселов и более приводит к сложным аналитическим вычислениям, но не дает существенного выигрыша в точности определения вероятности пропуска «чужого».

Приведем пример расчета вероятности пропуска «чужого» субъекта при разбиении изображения отпечатка пальца на сектора размером 3×3 пиксела.

Примем следующие исходные данные для расчета вероятности пропуска «чужого» субъекта:

– количество минуций, выделенных на предоставляемом субъектом изображении отпечатка пальца (p), примем равным 10;

– порог меры близости (g) примем равным 0,7 и 0,9.

Используя формулу (2.9), найдем значения количества минуций, выделенных на эталонном изображении отпечатка пальца (q), при которых не выполняется неравенство (2.12).

Полученные значения количества минуций, выделенных на эталонном изображении отпечатка пальца, представлены в табл. 2.2.

Таблица 2.2

Количество минуций, выделенных на эталонном изображении отпечатка пальца

Порог меры близости, g	Количество минуций, выделенных на эталонном изображении отпечатка пальца, q
0,7	7 – 14
0,9	9 – 11

По формуле (2.10) рассчитаем границы критических областей для количества совпавших пар минуций (D), а по формуле (2.14) – соответствующие им вероятности пропуска «чужого» субъекта ($P_{\text{ПГ}}$) для полученных значений количества минуций, выделенных на эталонном изображении отпечатка пальца. Полученные результаты вероятности пропуска «чужого» для p = 10 представлены в табл. 2.3.

Таблица 2.3

Допустимые значения количества совпавших пар минуций
и соответствующие им вероятности пропуска «чужого» субъекта для $p = 10$

Порог меры близости, g	Количество минуций, выделенных на эталонном изображении отпечатка пальца, q	Нижняя граница критической области количества совпавших пар минуций, $\text{int}(\sqrt{g \cdot p \cdot q}) + 1$	Верхняя граница критической области количества совпавших пар минуций, $\min(p, q)$	Допустимые значения количества совпавших пар минуций, D	Вероятность пропуска «чужого», $P_{\text{ПГ}}$
0,7	7	7	7	7	$9{,}198 \cdot 10^{-7}$
	8	8	8	8	$1{,}263 \cdot 10^{-7}$
	9	8	9	8, 9	$1{,}436 \cdot 10^{-7}$
	10	9	10	9, 10	$1{,}972 \cdot 10^{-8}$
0,7	11	9	10	9, 10	$1{,}972 \cdot 10^{-8}$
	12	10	10	10	$2{,}381 \cdot 10^{-9}$
	13	10	10	10	$2{,}381 \cdot 10^{-9}$
	14	10	10	10	$2{,}381 \cdot 10^{-9}$
0,9	9	9	9	9	$1{,}734 \cdot 10^{-8}$
	10	10	10	10	$2{,}381 \cdot 10^{-9}$
	11	10	10	10	$2{,}381 \cdot 10^{-9}$

2.5. Определение вероятности пропуска «чужого» субъекта средством аутентификации по образцу голоса

Выходной сигнал речевого тракта представляет собой свертку сигнала возбуждения и импульсного отклика голосового тракта

$$x(t) = e(t) \cdot h(t), \qquad (2.15)$$

где $x(t)$ – входной речевой сигнал;

$e(t)$ – сигнал возбуждения;

$h(t)$ – импульсный отклик голосового тракта

t – время.

Эту систему можно рассмотреть в частотной области, тогда преобразование Фурье речевого сигнала равно произведению преобразований Фурье функции возбуждения и импульсного отклика голосового тракта:

$$X(w) = E(w) \cdot H(w), \qquad (2.16)$$

где ω – частота.

Спектр периодической возбуждающей последовательности $E(\omega)$ является линейным, его гармоники отстоят друг от друга на $2\pi/T_{\text{в}}$, где $T_{\text{в}}$ – это период сигнала возбуждения. Частотная характеристика голосового тракта $H(\omega)$ является сравнительно гладкой функцией частоты. При создании различных звуков форма речевого тракта изменяется, изменяется при этом и форма огибающей спектра речевого сигнала во времени. Следовательно, чтобы проводить корректный спектральный анализ речи, следует иметь в виду

кратковременный спектральный анализ на интервале времени 10–30 мс, допуская, что за этот временной интервал речевой тракт не успевает существенно изменить свою геометрию за счет перестройки артикуляторов.

Таким образом, речевой сигнал удобнее представлять не в виде непрерывной функции времени *t*, а в виде одномерного оцифрованного сигнала *x*(*n*), где *n* – порядковый номер интервала времени.

В данном средстве аутентификации в качестве параметров, на основании которых происходит сравнение предоставленного субъектом образца речи с эталонным, выступают кепстральные коэффициенты оцифрованных сигналов этих образцов [41]:

$$c_h(m) = \frac{1}{L}\sum_{k=1}^{L} \lg\left|X_h(k)\right| e^{j\frac{2p}{L}km}; h = 1,2,...,T;\ m = 1,2,...,L, \qquad (2.17)$$

где $c_h(m)$ – h-й кепстральный коэффициент n-го интервала времени;

T – количество интервалов времени в оцифрованных сигналах, которые формируются методом векторного квантования «k-средних»;

L – количество кепстральных коэффициентов, вычисляемых для каждого интервала времени.

В данном средстве аутентификации используются три эталонные матрицы кепстральных коэффициентов S_1, S_2 и S_3 размерностями $T \times L$. Они создаются путем трехкратной записи субъектом своего звукового пароля.

Эталонные матрицы имеют следующий вид:

$$S_1 = \begin{bmatrix} c_1(1)_1 & c_2(1)_1 & \mathbf{L}\, c_T(1)_1 \\ c_1(2)_1 & c_2(2)_1 & \mathbf{L}\, c_T(2)_1 \\ \mathbf{L} & & \\ c_1(L)_1 & c_2(L)_1 & \mathbf{L}\, c_T(L)_1 \end{bmatrix}; \quad S_2 = \begin{bmatrix} c_1(1)_2 & c_2(1)_2 & \mathbf{L}\, c_T(1)_2 \\ c_1(2)_2 & c_2(2)_2 & \mathbf{L}\, c_T(2)_2 \\ \mathbf{L} & & \\ c_1(L)_2 & c_2(L)_2 & \mathbf{L}\, c_T(L)_2 \end{bmatrix}; \quad S_3 = \begin{bmatrix} c_1(1)_3 & c_2(1)_3 & \mathbf{L}\, c_T(1)_3 \\ c_1(2)_3 & c_2(2)_3 & \mathbf{L}\, c_T(2)_3 \\ \mathbf{L} & & \\ c_1(L)_3 & c_2(L)_3 & \mathbf{L}\, c_T(L)_3 \end{bmatrix}. \qquad (2.18)$$

Аналогично входная речевая последовательность методом векторного квантования преобразуется в текущую матрицу S размерностью $T \times L$:

$$S = \begin{bmatrix} c_1(1) & c_2(1) & \mathbf{L}\ c_T(1) \\ c_1(2) & c_2(2) & \mathbf{L}\ c_T(2) \\ \mathbf{L} & & \\ c_1(L) & c_2(L) & \mathbf{L}\ c_T(L) \end{bmatrix}. \qquad (2.19)$$

Для сравнения матрицы, сформированной по голосу субъекта, с эталонными формируются три матрицы мер близости D_1, D_2, D_3 размерностями $T \times L$, такие, что

$$d_1(n,z) = \sum_{m=1}^{L} w_m (S_{1n,m} - S_{z,m})^2;$$

$$d_2(n,z) = \sum_{m=1}^{L} w_m (S_{2n,m} - S_{z,m})^2; \qquad (2.20)$$

$$d_3(n,z) = \sum_{m=1}^{L} w_m (S_{3n,m} - S_{z,m})^2,$$

где w_m – обратное значение дисперсии m-го кепстрального коэффициента входной речевой последовательности;

n, z – номера интервалов времени эталонной и входной речевых последовательностей соответственно.

Далее из каждой матрицы выбираются T значений, которые соответствуют минимальным мерам близости так, что каждому столбцу и каждой строке матрицы может принадлежать только одно из выбранных значений. Окончательно мера близости d между предоставляемой матрицей и эталонной определяется как среднее арифметическое из T выбранных значений. Если для данной эталонной матрицы окончательная мера близости окажется меньше заданного порога меры близости g, то «чужой» субъект будет пропущен. В случае пропуска «чужого» субъекта по двум из трех матриц мер близости произойдет пропуск средством аутентификации «чужого» субъекта.

Значения матриц мер близости зависят от значений матриц кепстральных коэффициентов. В литературе приводятся гистограммы распределений кепстральных коэффициентов на множестве субъектов, которые имеют нормальный закон распределения с разными математическими ожиданиями. Математические ожидания большинства нормальных распределений кепстральных коэффициентов на множестве субъектов близки к нулю. Математические ожидания нормальных распределений первого и второго кепстральных коэффициентов на множестве субъектов смещены относительно нуля.

Исходя из описания средства аутентификации вероятность пропуска «чужого» – это вероятность того, что окончательная мера близости двух из трех матриц мер близости окажется меньше заданного порога меры близости. Таким образом вероятность пропуска «чужого» зависит от значений матриц мер близости:

$$P_{\text{ПГ}} = f(D_1, D_2, D_3, g). \qquad (2.21)$$

Законы распределения случайных величин значений матриц D_1, D_2, D_3 не установлены и не зафиксированы в литературе, поэтому для определения этих значений в каждом частном случае требуется производить длительные эксперименты и громоздкие аналитические расчеты. Так, для упрощения формулы (2.21) произведем нормирование матриц мер близости, в результате чего случайная величина значений этих матриц из непрерывной неограниченной реализации преобразуется в непрерывную реализацию на отрезке [0,1]. Для построения нормированных матриц мер близости Dn_1, Dn_2, Dn_3 воспользуемся следующей тригонометрической функцией:

$$f(x) = \frac{2}{p}\left|\text{arctg}(x)\right|. \qquad (2.22)$$

Нормирование с использованием данной тригонометрической функции, в силу ее свойств, позволяет выделить значения матриц мер близости, близких к нулю, которые влияют на определение субъекта «своим».

Тогда значение нормированных матриц мер близости будут вычисляться следующим образом:

$$\begin{aligned} dn_1(n,z) &= \frac{2}{p}\left|\text{arctg}(d_1(n,z))\right|; \\ dn_2(n,z) &= \frac{2}{p}\left|\text{arctg}(d_2(n,z))\right|; \\ dn_3(n,z) &= \frac{2}{p}\left|\text{arctg}(d_3(n,z))\right|. \end{aligned} \qquad (2.23)$$

Аналогичное нормирование необходимо произвести с порогом g. Нормированное значение порога меры близости Δ вычисляется по следующей формуле:

$$\Delta = \frac{2}{p}\left|\text{arctg}(g)\right|. \qquad (2.24)$$

Непрерывность реализации случайной величины значений нормированных матриц мер близости на отрезке [0, 1] предполагает бесконечное количество значений этой случайной величины. В связи с этим производится дискретизация нормированных значений матриц мер близости с шагом Δ. В результате дискретизации все значения нормированных матриц мер близости, меньшие Δ, становятся равными нормированному порогу меры близости.

Наличие в дискретной нормированной матрице мер близости значения, равного нормированному порогу меры близости, означает, что кепстральные коэффициенты временного интервала, номер которого равен номеру строки (эталонной речевой последовательности) соответствуют кепстральным коэффициентам временного интервала, номер которого равен номеру столбца (входной речевой последовательности). Если в дискретной нормированной матрице мер близости найдутся T таких значений, каждое из которых принадлежит своей строке и своему столбцу, то кепстральные коэффициенты всех временных интервалов эталонной речевой последовательности попарно будут соответствовать кепстральным коэффициентам временных интервалов входной речевой последовательности.

Следовательно, субъект будет признан «своим» по дискретной нормированной матрице мер близости, если в ней найдутся T значений, равных нормированному порогу меры близости, таких, что каждому столбцу и каждой строке матрицы будет принадлежать только одно из выбранных значений.

Для каждого из значений, равного Δ, из области поиска в нормированной дискретной матрице мер близости нужно последовательно исключать одну строку и один столбец, которым принадлежит ячейка с найденным значением.

Вероятность того, что «чужой» субъект будет пропущен по дискретной нормированной матрице мер близости, будет равна

$$P_{\Delta} = \prod_{i=T}^{1} P_{\Delta i}, \tag{2.25}$$

где i – переменная, значение которой равно количеству строк или столбцов квадратной матрицы; $i = T, T–1,\ldots, 1$;

$P_{\Delta i}$ – вероятность того, что найдется хотя бы одно значение дискретной нормированной матрицы мер близости размерностью $i \times i$ (с учетом использованных столбцов и строк) равное Δ. $P_{\Delta i}$, в свою очередь, равна

$$P_{\Delta i} = \sum_{h=1}^{i^2} \frac{(i^2)!}{h! \cdot (i^2 - h)!} \cdot \Delta^h \cdot (1-\Delta)^{(i^2-h)}, \tag{2.26}$$

где h – переменная, значение которой равно количеству ячеек в матрице размером $i \times i$; $h = 1, 2,\ldots, i^2$.

Тогда формулы вероятностей P_1, P_2, P_3 того, что «чужой» субъект будет пропущен по первой, второй или третьей дискретным нормированным матрицам мер близости соответственно, будут иметь вид

$$\begin{aligned} P_1 &= \prod_{i=T}^{1} \sum_{h=1}^{i^2} \frac{(i^2)!}{h! \cdot (i^2 - h)!} \cdot \Delta^h \cdot (1-\Delta)^{(i^2-h)}; \\ P_2 &= \prod_{i=T}^{1} \sum_{h=1}^{i^2} \frac{(i^2)!}{h! \cdot (i^2 - h)!} \cdot \Delta^h \cdot (1-\Delta)^{(i^2-h)}; \\ P_3 &= \prod_{i=T}^{1} \sum_{h=1}^{i^2} \frac{(i^2)!}{h! \cdot (i^2 - h)!} \cdot \Delta^h \cdot (1-\Delta)^{(i^2-h)}. \end{aligned} \tag{2.27}$$

В связи с тем, что размерности всех трех дискретных нормированных матриц мер близости одинаковы, вероятности пропуска, «чужого» субъекта по первой, второй или третьей дискретным нормированным матрицам мер близости, также будут одинаковы:

$$P_1 = P_2 = P_3 = P_{\Delta}. \tag{2.28}$$

Вероятность того, что случайная величина значений дискретных нормированных матриц мер близости будет иметь значение Δ из интервала [0, 1], равна Δ.

Вероятностью пропуска «чужого» субъекта для данного средства аутентификации является вероятность того, что в двух матрицах из трех дискретных нормированных матриц мер близости найдется T значений, равных нормированному порогу меры близости Δ, таких, что каждому столбцу и каждой строке данной матрицы будет принадлежать только одно из выбранных значений.

Таким образом, формула вероятности пропуска «чужого» субъекта имеет вид

$$P_{\text{ПГ}} = P_1 \cdot P_2 \cdot (1-P_3) + P_1 \cdot P_3 \cdot (1-P_2) + P_2 \cdot P_3 \cdot (1-P_1) + P_1 \cdot P_2 \cdot P_3. \tag{2.29}$$

Используя выражение (2.28), получим

$$\begin{aligned} P_{\text{ПГ}} &= P_1 \cdot P_2 \cdot (1-P_3) + P_1 \cdot P_3 \cdot (1-P_2) + P_2 \cdot P_3 \cdot (1-P_1) + P_1 \cdot P_2 \cdot P_3 = \\ &= 3 \cdot P_{\Delta}^{\,2} \cdot (1-P_{\Delta}) + P_{\Delta}^{\,3}, \end{aligned} \tag{2.30}$$

где вероятности P_1, P_2 и P_3 вычисляются по формуле (2.27).

Приведем пример расчета вероятности пропуска «чужого» субъекта данным средством аутентификации.

Примем следующие исходные данные:

– количество интервалов времени в тестируемой речевой последовательности и в эталонных записях $T = 32$;

– количество кепстральных коэффициентов (L), которые используются в процессе создания матриц S, S_1, S_2, S_3 примем равным 14;

– порог меры близости $d = 0{,}1$.

Согласно формуле (2.24) нормированный порог меры близости равен

$$\Delta = \frac{2}{p}\left|\operatorname{arctg}(0{,}1)\right| = 0{,}063.$$

Вероятность того, что «чужой» субъект будет пропущен по одной из трех дискретных нормированных матриц мер близости, равна

$$P_{\Delta} = \prod_{i=32}^{1}\sum_{j=1}^{i^2}\frac{(i^2)!}{j!\cdot(i^2-j)!}\cdot 0{,}063^{j}\cdot 0{,}937^{(i^2-j)} = 2{,}9\cdot 10^{-3}.$$

Тогда вероятность пропуска «чужого» субъекта равна

$$P_{\text{ПГ}} = 3\cdot(2{,}9\cdot 10^{-3})^2\cdot(1-2{,}9\cdot 10^{-3}) + (2{,}9\cdot 10^{-3})^3 = 2{,}52\cdot 10^{-5}.$$

Аналогичным образом можно определить аналитические выражения для расчёта вероятностей пропуска «чужого» субъекта другими биометрическими средствами аутентификации.

2.6. Принципы проектирования средств аутентификации

Вероятность правильного опознания субъекта средством аутентификации, как уже было рассмотрено в гл. 1, определяется по формуле

$$P_{\text{ПО}} = (1-P_{\text{ПА}})\cdot(1-P_{\text{ОТ}})\cdot(1-P_{\text{ДН}}).$$

Наличие компонент $(1–P_{\text{ОТ}})$ и $(1–P_{\text{ДН}})$ снижает значение вероятности правильного опознания по сравнению с верхней границей, определяемой компонентой $(1–P_{ПА})$.

Для того чтобы увеличить вероятность правильного опознания, необходимо минимизировать вероятности пропуска «чужого» в результате сбоев (отказов) оборудования и действий нарушителя. Этого можно достичь, используя при проектировании средств аутентификации следующие принципы:

– принцип максимального правдоподобия;

– принцип ограничения попыток;

– принцип цикличности.

Принцип максимального правдоподобия заключается в следующем. Пусть $A = \{a_i\}$, $i = 1, n$ – эталонные значения параметров, используемых для аутентификации, а $X = \{x_i\}$, $i = 1, n$ – значения параметров, предъявляемых для аутентификации.

Пусть независимые попытки аутентификации имеют частные вероятности $\rho(X,A)$, тогда принцип максимального правдоподобия состоит в

выборе в качестве истинного такого параметра X, при котором максимизируется функция правдоподобия:

$$L(q) = r(x_1, a_1), r(x_2, a_2) \mathbf{K}\, r(x_n, a_n).$$

Для средств аутентификации двух первых классов, в которых опознание происходит на основании того, «что знает субъект» и «что имеет субъект», принцип максимального правдоподобия заключается в том, что аутентификация считается установленной при абсолютном совпадении всех сравниваемых признаков входного (предоставленного субъектом) и эталонного (хранящегося в памяти средства аутентификации) воздействий. Это обусловлено тем, что результат преобразования признаков, предоставляемых одним и тем же субъектом, в понятный средству аутентификации вид всегда имеет одинаковые значения.

Увеличение вероятности правильного опознания субъекта для данных средств аутентификации достигается за счет расширения алфавита или длины аутентификатора.

Для биометрических средств аутентификации абсолютное совпадение всех сравниваемых признаков входного и эталонного воздействий недостижимо. Это обусловлено тем, что процесс преобразования признаков, предоставленных субъектом, в понятный средству аутентификации вид носит вероятностный характер. В этом случае принцип максимального правдоподобия заключается в том, что аутентификация считается установленной, если величина несовпадения всех сравниваемых признаков входного и эталонного воздействий, не превышает некоторого значения меры близости сравниваемых признаков.

Увеличение вероятности правильного опознания субъекта биометрическими средствами аутентификации достигается за счет минимизации значения меры близости сравниваемых признаков, что, с другой стороны, может привести к увеличению вероятности блокирования «своих» субъектов. Другим путем увеличения вероятности правильного опознания является максимизация алфавита биометрических признаков за счет изменения точности их получения и сравнения с эталонными. Например, для средства аутентификации по отпечатку пальца максимизацию алфавита биометрических признаков проводят за счет увеличения разрешения картинки отпечатка пальца, а для средства аутентификации по голосу – за счёт увеличения размера секторов, в которых происходит определение типа минуций.

Принцип ограничения попыток заключается в том, что при аутентификации субъекта ограничивается число попыток неправильного входа в систему.

При отсутствии ограничения на число попыток неправильного входа значение вероятности подбора аутентификатора определяется по формуле

$$P_{\text{ПА}} = P_{\text{ПА1}} + (1 - P_{\text{ПА1}}) \cdot P_{\text{ПА2}} + (1 - P_{\text{ПА1}}) \cdot (1 - P_{\text{ПА2}}) \cdot P_{\text{ПА3}} + \mathbf{L} + (1 - P_{\text{ПА1}}) \cdot \mathbf{L} \cdot (1 - P_{\text{ПА}N\text{-}1}) \cdot P_{\text{ПА}N},$$

где N – объем алфавита;

$P_{\text{ПА}i}$ – вероятность подбора аутентификатора с i-й попытки, равная

$$P_{\text{ПА}i} = \frac{1}{N - i + 1},\ i = 1, 2, \ldots, N.$$

При использовании принципа ограничения попыток вероятность подбора аутентификатора за k попыток будет равна

$$P_{\text{ПА}} = \frac{1}{N} + (1-\frac{1}{N}) \cdot \frac{1}{N-1} + \mathbf{L} + (1-\frac{1}{N}) \cdot \mathbf{L} \cdot (1-\frac{1}{N-k+2}) \cdot \frac{1}{N-k+1} = \frac{k}{N}.$$

В связи с тем, что $k << N$, при использовании данного принципа вероятность того, что субъект не подберет аутентификатор за k попыток незначительно уменьшается по отношению к верхней границе вероятности правильного опознания субъекта и не изменяется при превышении количества попыток величины k (рис. 2.11).

Данный принцип реализуется путем подсчета неправильных вводов аутентификатора и блокировки средства аутентификации при превышении допустимого количества попыток неправильного входа в систему.

В разработанной модели средств аутентификации предусмотрена возможность блокировки после превышения допустимого количества попыток неправильного входа в систему. В данном случае на вход средства аутентификации подается разрешенное входное воздействие и устанавливается событие превышения допустимого количества попыток неправильного входа в систему. Вероятность появления на выходе разрешающего выходного воздействия при этих условиях обусловливается лишь сбоем (отказом) оборудования и действиями нарушителя.

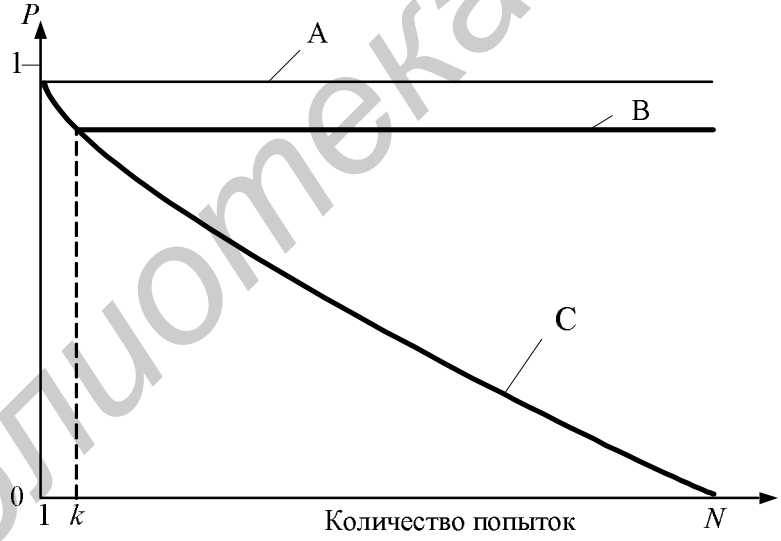

Рис. 2.11. Зависимость вероятности правильного опознания субъекта от количества попыток входа в систему:
A – верхняя граница вероятности правильного опознания;
B – вероятность того, что субъект не подберет аутентификатор за k попыток;
C – вероятность того, что субъект не подберет аутентификатор за N попыток

Принцип цикличности заключается в том, что средство аутентификации функционирует по заранее установленному жесткому циклу, и ни при каких входных воздействиях последовательность выполнения его функций не нарушается.

Данный принцип является следствием утверждений 1.3 и 1.4 (подгл.1.4), в связи с чем является стержневым при построении алгоритма работы средства аутентификации.

При использовании данного принципа в качестве параметра, учет которого позволяет уменьшить вероятность пропуска «чужого» субъекта, выступает безопасное время действия аутентификатора $T_б$, связанное с вероятностью подбора аутентификатора простым соотношением

$$T_б = \frac{P_{ПАk}}{P_{ПА}} T_ц = N \cdot P_{ПАk} \cdot T_ц ,$$

где $T_б$ – безопасное время действия аутентификатора;

$P_{ПАk}$ – вероятность подбора аутентификатора за k попыток, которые возможно реализовать за время $T_б$;

$T_ц$ – время выполнения средством аутентификации одного цикла работы.

В силу того что цикл работы жестко фиксирован, введя некоторую временную задержку в конце цикла, можно существенно повысить безопасное время аутентификатора при фиксированной вероятности подбора аутентификатора за k попыток, которые возможно реализовать за время $T_б$. В данном случае безопасное время действия аутентификатора определяется как

$$T_б = \frac{P_{ПАk}}{P_{ПА}} \cdot (T_ц + t_з) = N \cdot P_{ПАk} \cdot (T_ц + t_з) ,$$

где $t_з$ – временная задержка.

Тогда

$$P_{ПАk} = \frac{T_б}{N \cdot (T_ц + t_з)} .$$

Отсюда следует, что при фиксированных значениях времени безопасного использования аутентификатора и времени выполнения средством аутентификации цикла работы вероятность пропуска «чужого» для средства аутентификации, в котором отсутствует данная временная задержка, будет больше, чем для средства аутентификации, использующего временную задержку.

На основании рассмотренных выше принципов сформулируем требования, предъявляемые к средствам аутентификации при их проектировании.

Требование 1. В функциональной структуре средства аутентификации должны присутствовать функции обнаружения, опознания, управления и контроля.

Требование 2. Средство аутентификации в процессе функционирования должно обеспечивать выполнение функций обнаружения, опознания, управления и контроля в строгой последовательности.

Требование 3. Средство аутентификации должно полностью выполнять цикл указанных функций, независимо от результата работы каждой из них.

Требование 4. Функция управления средства аутентификации должна включать этап подсчета количества попыток неправильного входа в систему.

Допустимое количество таких попыток должно быть ограничено. Должна быть исключена также возможность удаления или изменения статистики количества данных попыток.

Требование 5. При отработке запрещенного входного воздействия или при превышении допустимого количества попыток неправильного входа в цикл работы средства аутентификации должна вноситься существенная временная задержка.

Требование 6. При отсутствии входного воздействия на выходе средства аутентификации должно быть установлено запрещающее выходное воздействие.

Требование 7. Во время цикла обработки входного воздействия входные интерфейсы средства аутентификации должны быть заблокированы, чтобы исключить возможность приема на обработку других входных воздействий. Разблокировка входных интерфейсов должна осуществляться только после завершения цикла работы, генерации и выдачи выходного воздействия.

Требование 8. Должна быть исключена возможность вывода из строя средства аутентификации путем воздействия на его органы управления или подачи каких-либо входных воздействий.

Требование 9. Разрешающее и запрещающее выходные воздействия, генерируемые средством аутентификации, должны обеспечивать их однозначную идентификацию в условиях возможных помех или сбоев оборудования.

Требование 10. Информационные и предписывающие сообщения, выдаваемые средством аутентификации субъекту, должны исключать возможность их неоднозначной интерпретации.

Требование 11. Должна быть исключена возможность ознакомления, удаления или изменения секретных данных субъектов, хранящихся в памяти средства аутентификации.

Требование 12. Диалог между средством аутентификации и устройством ввода аутентификатора должен быть реализован таким образом, чтобы исключить компрометацию секретных параметров.

Требование 13. Передача аутентификатора из электронного носителя в средство аутентификации должна осуществляться только после установления их взаимного соответствия.

Требование 14. Входной интерфейс средства аутентификации должен иметь быстродействие, обеспечивающее заданную вероятность правильного опознания в течение безопасного времени использования аутентификатора.

2.7. Методика оценки средства аутентификации

Методика оценки средства аутентификации включает в себя следующие этапы:

1) формализация исследуемого средства аутентификации;

2) определение цикличности функциональной структуры средства аутентификации;

3) оценка вероятностей событий, приводящих к нарушению работы;

4) исследование параметров средства аутентификации на модели и оценка результатов исследования.

Этап 1. Формализация исследуемого средства аутентификации состоит из следующих действий:

– определяется перечень всех входных и выходных интерфейсов средства аутентификации;

– для каждого интерфейса определяются элементы, осуществляющие реализацию функций обнаружения, опознания, управления и контроля;

– для каждой из функций устанавливается взаимно однозначное соответствие выходных сигналов, команд, параметров элементов входным сигналам, командам, параметрам элементов сопрягаемых функций;

– для каждой функции устанавливается возможность формирования команд, сигналов перехода средства аутентификации в состояние выполнения предыдущей функции;

– определяются элементы, осуществляющие формирование сигналов (команд) управления и выявляются возможные пути обхода этих элементов, приводящие к формированию сигнала «Разрешение».

Этап 2. Для определения того, что функции обнаружения, опознания, управления и контроля выполняются последовательно и составляют цикл работы, устанавливается, что:

– выходные сигналы, сформированные при завершении выполнения каждой из функций, являются входными сигналами только для соответствующих элементов сопрягаемых функций;

– сигналы (команды) на перевод средства аутентификации в исходное состояние формируются только после завершения выдачи им выходных сигналов, являющихся результатом обработки входного воздействия;

– средство аутентификации переводится в состояние разрешения начала обработки следующего из подлежащих анализу воздействий только после выдачи на выходной интерфейс сигналов разрешения или запрета доступа.

Этап 3. Для ситуаций, рассмотренных на этапе 2, определяют оценки вероятностей следующих событий:

– отказ (сбоя) элементов, осуществляющих выполнение каждой из функций, приводящего к переходу средства аутентификации из запрещающего состояния в разрешающее;

– нарушения взаимно однозначного соответствия выходных сигналов, команд, параметров элементов для каждой из функций входным сигналам, командам, параметрам элементов сопрягаемых функций;

– обход элементов, реализующих выполнение каждой из функций, по каждому виду входного воздействия;

– изменение нарушителем алгоритмов выполнения функций, приводящее к переходу средства аутентификации в разрешающее состояние;

– изменение нарушителем секретных данных, хранящихся в средстве аутентификации и влияющих на правильность опознания.

Этап 4. Исследование параметров средства аутентификации на модели и оценка результатов исследования включает в себя:

– выбор исходных данных для работы программы модели средств аутентификации (тип средства аутентификации, вид входного воздействия, вероятности враждебных переходов);

– определение разрешенных и запрещенных состояний функций обнаружения, опознания, управления и контроля средства аутентификации;

– формирование матрицы переходов между состояниями средства аутентификации с использованием определённых на этапе 3 вероятностей;

– расчет с помощью программы модели средств аутентификации вероятности пропуска «чужого» средством аутентификации с учётом отказов;

– расчет с помощью программы модели средств аутентификации вероятности пропуска «чужого» средством аутентификации с учётом действий нарушителя.

– определение верхней границы вероятности правильного опознания субъекта исследуемым средством аутентификации;

– определение вероятности правильного опознания субъекта средством аутентификации при заданных условиях.

Контрольные вопросы и задачи

1. Постройте граф модели функции опознания средства аутентификации с использованием паролей в ОС UNIX, если известно, что в отличие от ОС Windows, в UNIX ввод пароля осуществляется после проверки имени пользователя.

2. Как изменится граф модели функции контроля, если в средстве аутентификации отсутствует режим блокировки?

3. Вычислите вероятность пропуска «чужого» средством аутентификации по отпечатку пальца при разбиении его изображения на сектора размером 3×3 пиксела и пороге меры близости 0,7.

4. Поясните, зачем в устройстве аутентификации по образцу голоса используется три матрицы мер близости?

5. Назовите принципы проектирования средств аутентификации и поясните их смысл.

6. Проведите сравнительные расчёты времени безопасного действия пароля при наличии и отсутствии времени задержки по циклу после неправильно введённого пароля. Какой, по вашему мнению, должна быть оптимальная задержка времени по циклу?

ГЛАВА 3. ПРОТОКОЛЫ АУТЕНТИФИКАЦИИ

Первой задачей протоколов аутентификации является установление подлинности участников взаимодействия, разделённых коммуникационной системой. Все протоколы аутентификации включают как минимум двух участников:

1) *A – доказывающего* – участника, проходящего аутентификацию;
2) *B – проверяющего* – участника, проверяющего аутентичность доказывающего.

Целью протокола является проверка того, что проверяемым действительно является *A*.

С точки зрения проверяющего возможными исходами протокола являются либо принятие решения об идентичности доказывающего *A*, либо завершение протокола без принятия такого решения.

Как правило, в ходе протокола аутентификации подлинность участников взаимодействия выясняется путём определения подлинности соответствующих сообщений. Поэтому, чтобы убедиться в подлинности сообщения и его автора, используются механизмы аутентификации источника данных, предусматривающие выполнение следующих действий:

1) передача сообщения от отправителя к получателю, проверяющему достоверность сообщения до его обработки;
2) идентификация отправителя сообщения;
3) проверка целостности данных, полученных от отправителя;
4) проверка подлинности отправителя сообщения.

Следующая задача протоколов аутентификации – обеспечить согласование ключей шифрования, используемых для защиты передаваемой информации. Как правило, стороны, обменивающиеся информацией, запускают протокол аутентификации для того, чтобы в дальнейшем перевести общение на более высокий уровень. В современной криптографии в основе организации защищенных каналов связи лежат криптографические ключи. Поэтому протоколы аутентификации для дальнейшего обмена информацией по защищенным каналам в качестве составной части содержат или механизм формирования аутентифицированных ключей, или механизм обмена ключами (согласования).

Существует много методов реализации протоколов аутентификации, однако к основным протокольным конструкциям, принятым в качестве международных стандартов, относятся следующие:

– механизмы определения «актуальности» сообщения и существования пользователя;
– односторонняя и взаимная аутентификация;
– аутентификация с привлечением доверенного посредника.

Проверка «актуальности» сообщения – неотъемлемая часть аутентификации источника данных, в процессе которой пользователь должен активно обмениваться

информацией с подлинным партнером. Для её реализации используются механизмы «запрос-ответ» и «метка времени».

В механизме «запрос-ответ» проверяющая сторона получает комбинацию, состоящую из протокольного сообщения и криптографической операции, выполненной доказывающей стороной таким образом, чтобы проверяющая сторона могла убедиться в ее существовании, проверив «актуальность» полученной информации. Как правило, проверяющая сторона получает криптографическое преобразование заранее сгенерированного ею же одноразового случайного числа. Если обратное преобразование правильно восстанавливает случайное число, то проверяющая сторона имеет основания считать, что доказывающая сторона действительно зашифровала его после получения запроса. Если интервал между запросом и ответом достаточно мал, сообщение считается «актуальным». Поскольку случайное число, посланное проверяющей стороной, было извлечено из достаточно большого пространства, нет никакой возможности предсказать его заранее.

Используя механизм метки времени, доказывающая сторона указывает время создания своего сообщения, используя криптографическую операцию. Следовательно, время создания сообщения становится неотъемлемой частью сообщения. После расшифрования сообщения проверяющая сторона может сравнить извлеченную метку времени со своим собственным временем (предполагается, что участники протокола используют общемировое стандартное время, например, по Гринвичу). Если разница во времени относительно мала, сообщение считается «актуальным».

Указанные механизмы проверки «актуальности» сообщения и существования пользователя обеспечивают только так называемую «одностороннюю аутентификацию», в которой аутентифицировался только один из двух участников протокола. При взаимной аутентификации пользователи аутентифицируют друг друга. Следует отметить, что взаимная аутентификация не является двукратной односторонней аутентификацией, а представляет собой специальную конструкцию, т.е. для правильной взаимной аутентификации дважды выполнить протокол односторонней аутентификации в противоположных направлениях недостаточно.

При функционировании открытых систем пользователи взаимодействуют в интерактивном режиме, причём если два незнакомых пользователя захотят установить между собой секретную связь, они должны аутентифицировать друг друга и организовать защищенный канал связи. Как правило, для аутентификации и формирования ключей в открытых системах используется централизованная служба аутентификации, известная под названием «доверенный посредник». Считается, что между доверенным посредником и большим количеством пользователей в системе существуют долговременные отношения. Он поддерживает базу данных, содержащую имена обслуживаемых клиентов, и может идентифицировать пользователя на основе криптографического ключа, заранее разделенного между ним и пользователем. Доверенный посредник является особым пользователем, которому доверяют

все остальные пользователи (клиенты), т.е. он всегда отвечает на запрос клиента точно в соответствии со спецификацией протокола и не выполняет никаких других действий, которые могли бы непреднамеренно снизить степень безопасности (например, открыть третьей стороне секретные ключи, разделенные между ним и его клиентами).

Для обеспечения аутентификации в сетях, основанных на телефонных линиях (коммутируемые или выделенные каналы, сети ISDN), используется ряд протоколов, работающих совместно с протоколом «точка – точка» (Point-to-Point Protocol – PPP) [RFC-1661]. Рассмотрение указанного протокола выходит за рамки данной книги, т.к. он не относится собственно к протоколам аутентификации, однако для обеспечения единой базы укажем структуру (формат) кадра PPP (рис. 3.1).

Байт фланга	Байт адреса	Контрольный байт	Два байта протокола	Данные	Контрольная последовательность кадров	Байт фланга

Рис. 3.1. Структура кадра PPP

При описании схем аутентификации в PPP используется следующая терминология: субъект, для которого требуется аутентификация, называется узлом, а объект, производящий аутентификацию – аутентификатором.

3.1. Протокол аутентификации по паролю

Данный протокол – наиболее простой из протоколов подтверждения удаленным субъектом своей подлинности объекту, предоставляющему ресурсы для использования. Аутентификация происходит за две итерации. При использовании протокола аутентификации по паролю (Password Authentication Protocol – PAP [RFC-1334]) в поле **Протокол** (два байта протокола) кадра PPP указывается соответствующее PAP-значение 0xC023, а поле данных преобразуется в четыре дополнительных поля (рис. 3.2).

Код	Идентификатор	Длина	Данные

Рис. 3.2. Структура поля **Данные кадра PPP**

При этом поле **Код** указывает на следующие возможные типы PAP-пакета:

Код = 1: Аутентификационный запрос.

Код = 2: Подтверждение аутентификации.

Код = 3: Отказ в аутентификации.

Поле **Идентификатор** обеспечивает соответствие пары «запрос-ответ» (должен меняться при каждом новом аутентификационном запросе).

Поле **Длина** указывает совокупную длину всех четырех полей.

Поле **Данные** содержит данные пакета – для аутентификационного запроса оно будет иметь вид, представленный на рис. 3.3.

Длина идентификатора	Идентификатор	Длина пароля	Пароль

Рис. 3.3. Структура поля **Данные PAP пакета-запроса**

Пакет аутентификационного запроса будет посылаться субъектом, желающим получить доступ неоднократно до наступления одного из следующих событий:

– до получения подтверждения или отказа;

– до истечения счетчика попыток.

При получении объектом запроса производится распознавание полученных результатов (сравнение с имеющимися у объекта значениями). По результатам распознавания субъекту высылается пакет с полем **Данные** следующего формата (рис. 3.4).

Длина сообщения	Сообщение

Рис. 3.4. Структура поля **Данные PAP пакета-ответа**

При этом в полях (см. рис. 3.2.) Код равен 2 или 3 (в зависимости от того, подтверждена аутентификация или отвергнута), в поле **Идентификатор** – идентификатор соответствующего запроса. В полях ответа (см. рис. 3.4) указывается: **Длина сообщения** – размер следующего поля, **Сообщение** – возлагается на конкретную реализацию (оно обязано не влиять на работу протокола, его рекомендовано формировать удобным для прочтения).

В протоколе дополнительно указано, что поскольку пакет с подтверждением аутентификации может быть утерян, реализация должна предусматривать возможность обработки повторного запроса на аутентификацию.

Таким образом, схема работы протокола имеет вид, представленный на рис. 3.5.

1. Устанавливается PPP-соединение.
2. Субъект посылает аутентификационный запрос с указанием своего идентификатора и пароля.
3. Объект проверяет полученные данные и подтверждает аутентификацию или отказывает в ней.

Следует обратить особое внимание, что весь обмен данными (в том числе и пересылка пароля) происходит в открытом виде, без применения криптографических средств. При этом частоту и время отправки пакетов контролирует сам субъект.

Для данного протокола существует другая известная реализация – SPAP (Shiva Password Authentication Protocol).

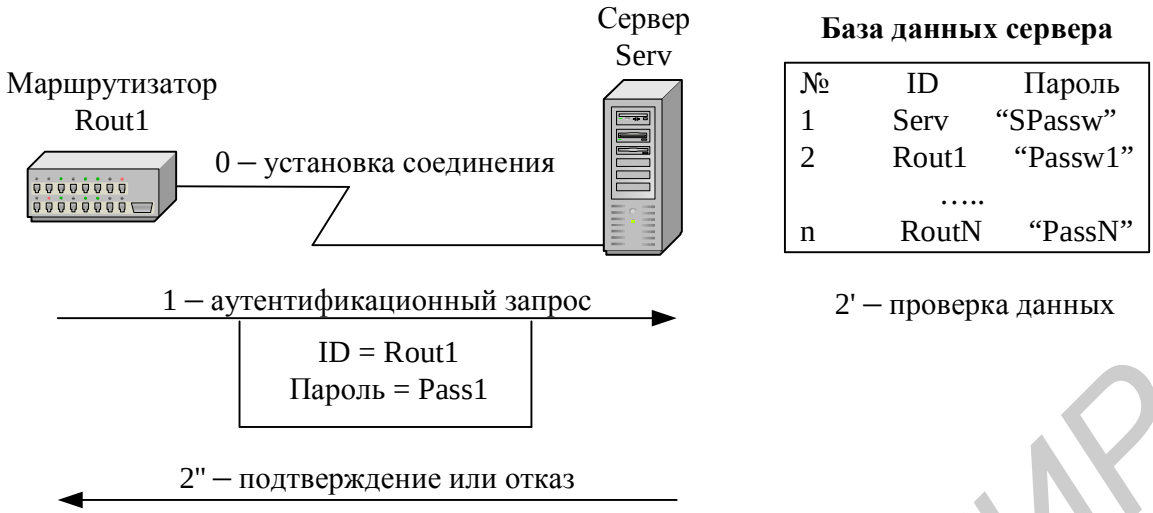

Рис. 3.5. Пример PAP-аутентификации маршрутизатора сервером

3.2. Протокол рукопожатия

Протокол рукопожатия (Challenge Handshake Authentication Protocol – CHAP [RFC-1994]) используется для первоначальной аутентификации субъекта после установления соединения, но может, в зависимости от конкретной реализации, быть использован для периодического подтверждения аутентифицированности субъекта во время работы в рамках установленного соединения. Аутентификация происходит за три итерации. В поле Протокол PPP-кадра указывается значение 0xC223, поле **Данные** преобразуется в четыре поля, аналогично PAP-пакету (см. рис. 3.2) со схожими типами. Единственное отличие между протоколами заключается в том, что поле Код имеет теперь четыре значения:

Код = 1: Запрос на предоставление данных для аутентификации.

Код = 2: Ответ на запрос с аутентификационными данными.

Код = 3: Подтверждение аутентификации.

Код = 4: Отказ в аутентификации.

Формат поля **Данные** зависит от поля **Код**.

Реализация протокола CHAP требует, чтобы обе стороны имели в распоряжении заранее согласованные секретные данные, которые не высылаются по сети, но чаще всего присутствуют у объекта в открытом виде. При этом объём секретных данных должен быть не меньше 1 байта, и в принципе его размер должен соответствовать требованиям хэш-алгоритма, выбранного для реализации.

Собственно процедура аутентификации инициируется в отличие от PAP не субъектом, а объектом и происходит следующим образом (рис. 3.6.):

1) устанавливается PPP-соединение;

2) объект посылает субъекту запрос на предоставление данных для аутентификации;

3) субъект отвечает необходимыми данными, часть из которых взята из содержимого запроса;

4) объект анализирует полученные данные и отвечает подтверждением или отказом.

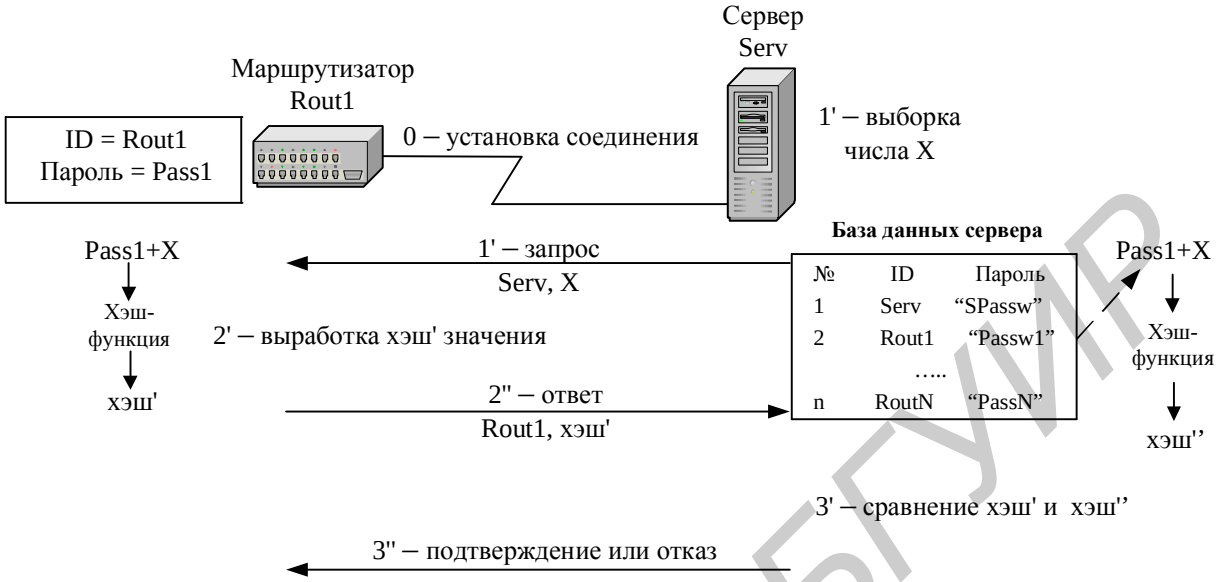

Рис. 3.6. Пример CHAP-аутентификации маршрутизатора сервером

Рассмотрим подробнее структуру пакета для запроса и ответа (поле Данные кадра PPP) (рис. 3.7).

Код	Идентификатор	Длина	Длина числа	Число	Имя

Рис. 3.7. Структура CHAP-пакета запроса или ответа

В поле **Код** указываются значения 1 для запроса и 2 для ответа.

Поле **Идентификатор** используется аналогично полю **Идентификатор** протокола PAP для обеспечения однозначного соответствия между запросом и ответом. Оно должно уменьшаться при новых запросах.

Поле **Длина** показывает суммарную длину всех полей пакета.

Поле **Длина** числа определяет размер следующего поля.

В поле **Число** указывается специальное значение, которое должно удовлетворять двум требованиям: быть уникальным и непредсказуемым. От качества выбора этого числа зависит, сможет ли злоумышленник использовать перехваченные им запросы-ответы при аутентификации уполномоченного субъекта. Для пакета **Запрос** это значение должно изменяться при каждом новом пакете.

В поле **Имя** указывается наименование системы, пославшей пакет. Так как субъект может аутентифицироваться на нескольких объектах, и объект

может аутентифицировать несколько субъектов, это поле можно использовать для поиска секретных данных в соответствующей базе данных.

После получения запроса субъект производит следующие действия:

1) выбирает соответствующие объекту секретные данные;

2) использует секретные данные и полученное **Число** как параметры для хэш-функции и получает соответствующее хэш-значение;

3) формирует пакет ответа, где в поле **Число** указывается хэш-значение, а в поле **Имя** – свое наименование;

4) высылает пакет объекту.

Объект, получив пакет, производит свое вычисление хэш-значения с использованием соответствующих секретных данных для данного субъекта и числа. В случае совпадения рассчитанного и присланного хэш-значения, происходит подтверждение аутентификации, иначе – отказ. Формат пакета для этого аналогичен формату пакета PAP с Кодом = 3 для успешной авторизации и с Кодом = 4 для отказа.

3.3. Расширенный протокол рукопожатия

Главная особенность расширенного протокола рукопожатия **(**Extensible Authentication Protocol – EAP [RFC-2284]) в том, что он может использовать различные (множественные) аутентификационные механизмы, при этом выбор механизма аутентификации переносится на усмотрение объекта, который может запросить у субъекта дополнительные параметры для определения такого механизма.

В поле **Протокол** PPP кадра указывается значение 0xC227, поле **Данные** преобразуется в четыре поля, аналогично PAP- и CHAP-пакету (рис. 3.2.) со схожими типами, единственное его отличие от других протоколов в том, что поле **Код** имеет четыре значения, как в пакете CHAP.

Код = 1: Запрос на предоставление данных.

Код = 2: Ответ на запрос.

Код = 3: Подтверждение аутентификации.

Код = 4: Отказ в аутентификации.

Поскольку процедура аутентификации может потребовать участия пользователя, необходимо предусматривать соответствующие тайм-ауты при рассылках объектом запросов на аутентификацию.

Процедура аутентификации инициируется объектом и происходит следующим образом.

1. Устанавливается PPP-соединение.

2. Объект посылает субъекту запрос на предоставление данных для аутентификации.

3. Субъект отвечает необходимыми данными, часть из которых взята из содержимого запроса.

4. Объект анализирует полученные данные и отвечает подтверждением или отказом.

Структура EAP пакета для запроса и ответа приведена на рис. 3.8. В поле **Код** указываются значения 1 для запроса и 2 – для ответа.

Код	Идентификатор	Длина	Тип	Данные типа

Рис. 3.8. Структура EAP-пакета запроса или ответа

Поле **Идентификатор** используется аналогично соответствующему полю протоколов PAP и CHAP для обеспечения соответствия между запросом и ответом. Оно должно уменьшаться при новых запросах. Однако необходимо учесть, что для рассмотренных случаев запаздывания с получением данных от пользователей поле Идентификатор может содержать повторяющееся значение (и должно совпадать для повторяющихся запросов).

Поле **Длина** указывает суммарную длину всех полей пакета.

Поле **Длина числа** указывает размер следующего поля.

Поле **Тип** обозначает тип запроса или ответа (собственно тип аутентификации) и должно указываться в запросе и совпадать с ним в ответе (если субъект поддерживает предложенный тип аутентификации), либо в ответе указывается NAK, если данный тип аутентификации не поддерживается. Субъект также может указать приемлемый для себя тип аутентификации).

Поле **Данные Типа** содержит данные, соответствующие указанному типу.

При успешной аутентификации объект высылает субъекту пакет следующей структуры (рис. 3.9).

Код	Идентификатор	Длина

Рис. 3.9. Структура EAP-пакета подтверждения или отказа аутентификации

Поле **Код** равно 3 при подтверждении и 4 – при отказе. Однако при этом объект может запросить дополнительную аутентификацию, учитывая, что отказ мог произойти по причине ошибки.

Поле **Идентификатор** соответствует идентификатору запроса, на который присылается ответ.

Поле **Длина** равно 4.

Теперь рассмотрим, какие типы аутентификации (значение поля **Тип** в запросе/ответе) поддерживаются протоколом.

Тип = 1: Идентификатор субъекта.

Тип = 2: Сообщение.

Тип = 3: NAK (только для ответов).

Тип = 4: Ответ MD5.

Тип = 5: Одноразовый пароль.

Тип = 6: Вид карты токена (Generic token card).

Тип 1 используется для определения субъекта, в поле **Данные Типа** возможно использование сообщения для приглашения пользователя ко вводу своего идентификатора.

Тип 2 используется для передачи субъекту сообщения (например, об истечении времени действия пароля).

Тип 3 используется только при ответе и указывает, что предложенный метод аутентификации неприемлем для субъекта, в поле **Данные Типа** указывается желаемый тип аутентификации.

Тип 4 используется для запроса/ответа (число или хэш-значение) согласно протоколу CHAP.

Тип 5 используется для запроса с одноразовым паролем.

Тип 6 используется для применения различных типов карточек. В запросе указывается текстовая (ASCII) информация, а в ответе – аутентификационные данные (например, то, что считано пользователем с карточки).

3.4. Протокол одноразовых паролей

Протокол одноразовых паролей **S/Key** предназначен для защиты от случаев, когда злоумышленник «прослушивает» сеть, пытаясь перехватить пароль (или соответствующее ему выражение, например, хэш-значение) для дальнейшего его использования. В системе S/Key [RFC-1760] каждый пароль пересылается по сети только однократно и после этого больше не используется, что делает описанную атаку бессмысленной. Пароль, вводимый собственно самим пользователем в интерфейсе программы, назовем *секретом*, чтобы не путать его далее с самим одноразовым паролем.

Система одноразовых паролей основана на клиент-серверном подходе. Клиент генерирует одноразовый пароль по определённой схеме, а сервер верифицирует его.

Использование одноразового пароля происходит в три фазы:

1) подготовительная (сбор данных для ввода);
2) рабочая (многократное применение хэш-функции к данным);
3) вывод (64-битовый одноразовый пароль выводится в виде, удобном для восприятия пользователем).

Первоначально клиент и сервер должны быть сконфигурированы для использования единого секрета, т.е. он должен присутствовать и у клиента, (например, в памяти пользователя), и у сервера. Далее, для создания уникальности одноразового пароля клиент и сервер должны определить случайное число и число итераций применения хэш-функции. Эти значения сервер в ответ на запрос об аутентификации (пакет инициализации) может выслать клиенту в открытом виде, они не являются секретными. При этом необходимо учесть, что одноразовые пароли используются сериями, в каждой серии случайное число – единое для каждого одноразового пароля, а число итераций уменьшается на 1 с каждым случаем использования пароля.

Рассмотрим пример такой серии, когда клиент очередной раз пытается получить доступ к серверу (рис. 3.10).

Клиент складывает полученное число (123) со своим секретом («Passw») и применяет к имеющемуся значению хэш-функцию столько раз, сколько указано в числе итераций (в нашем примере – 99). Полученный 64-битный результат и будет представлять собой одноразовый пароль. Поскольку его ввод требует участия пользователя, данный пароль представляется в виде шести блоков по 11 бит и заменяется шестью короткими английскими словами (от 1 до 4 букв) из фиксированного словаря в 2048 слов.

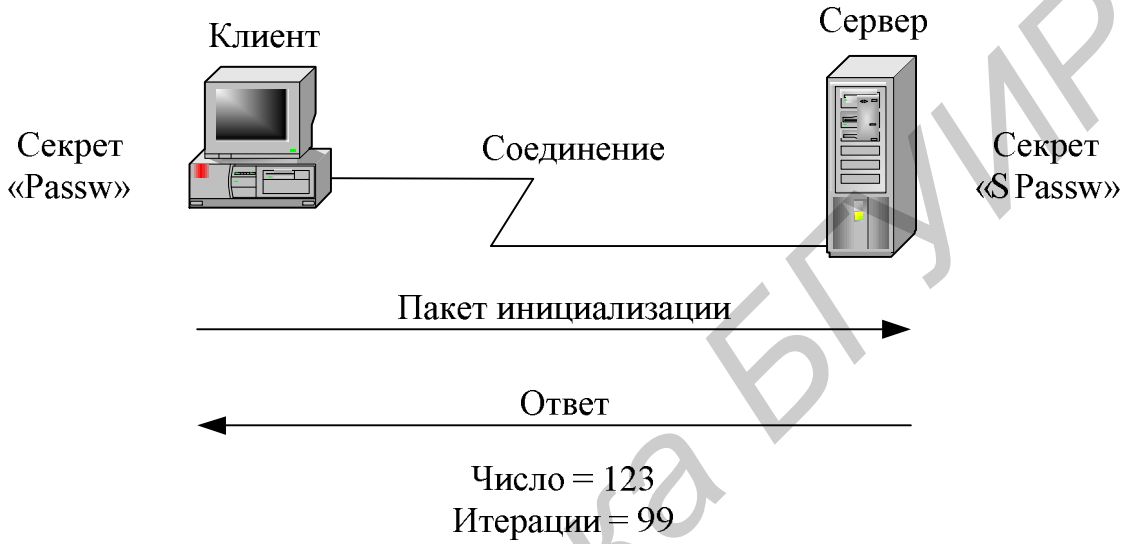

Рис. 3.10. Подготовительная фаза

Сервер хранит у себя последний успешный пароль серии, т. е. пароль, с которым клиент последний раз получал доступ к серверу. Теперь серверу необходимо лишь один раз применить к полученному от клиента паролю хэш-функцию и сравнить полученное значение с хранящимся у него последним успешным паролем. Если они совпадают – аутентификация прошла удачно (рис. 3.11). Предполагая, что используемая хэш-функция необратима, злоумышленник, даже перехватив текущий одноразовый пароль (например, со 100 итерациями), не может предугадать следующий пароль серии (с 99-ю итерациями).

По истечении количества итераций и клиенту, и серверу необходимо обновить случайное число, количество итераций и, возможно, секрет.

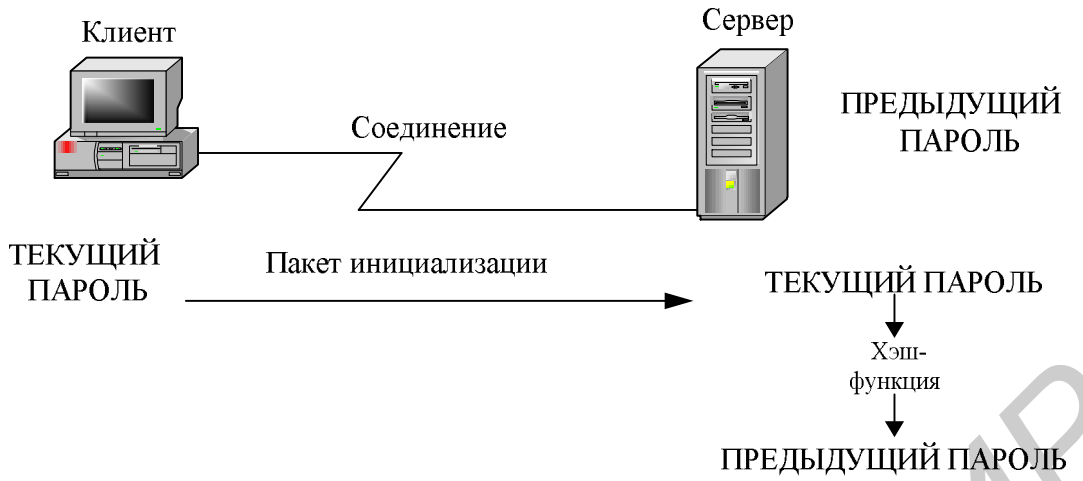

Рис. 3.11. Верификация

3.5. Протокол удалённой аутентификации при коммутируемом доступе

В качестве транспорта протокол удалённой аутентификации при коммутируемом доступе (Remote Authentication Dial-In User Service – RADIUS [RFC-2865, 2138]) использует UDP (в документе указаны четыре причины использования UDP вместо TCP):

– возможность использования вторичного сервера при сбое первичного;

– отличие в требованиях к синхронизации;

– отсутствие контроля состояния;

– обеспечение простоты при использовании сервера.

Это клиент-серверный протокол, где в качестве клиента обычно выступает сервер сетевого доступа (Network Access Server – NAS), с которым соединяется субъект-пользователь, а в качестве сервера – объект (сервер авторизации). RADIUS аутентифицирует транзакции на основе общего **Секрета** между ним и NAS, не передаваемого по сети, и шифрует этим же секретом пароли пользователей при пересылке между клиентом и сервером.

Один RADIUS-сервер может выступать в качестве посредника между клиентом и другим RADIUS-сервером, т.е. направлять второму серверу запросы клиента и возвращать клиенту ответы второго сервера.

Когда пользователь пытается получить доступ к объекту через клиента (в нашем случае – NAS), который использует для аутентификации RADIUS-сервер, то пользователь должен предоставить данному клиенту необходимую аутентификационную информацию. Клиент формирует **Запрос** на доступ, содержащий имя пользователя, его пароль, идентификатор клиента и идентификатор порта, к которому пользователь пытается получить доступ. При этом пароль скрывается путем шифрования алгоритмом MD5 в режиме обратной связи по шифртексту (CFB) с использованием в качестве начального вектора (IV) уникального номера **Запроса**, а в качестве ключа – заранее установленного между NAS и RADIUS **Секрета**.

Данный **Запрос** направляется RADIUS-серверу, причем повторяется несколько раз, если ответ от сервера не приходит в заданное время. Запросы могут отправляться другим серверам, если первичный сервер дал сбой.

При получении **Запроса** сервером он идентифицирует клиента и не обрабатывает **Запросы** от клиентов, с которыми у него нет общего **Секрета**. Если клиент одобрен, происходит анализ имени пользователя, указанного в **Запросе**. Сервер просматривает базу данных пользователей на предмет информации о данном пользователе, (например, о его пароле, идентификаторе клиента и порте, с которым ему разрешено работать). При этом сервер может обратиться к другим серверам за необходимой информацией (в этом случае он сам выступает в роли RADIUS-клиента). Если же в **Запросе** присутствуют атрибуты, указывающие, что RADIUS-сервер должен выступить в качестве посредника (proxy), то вся информация из **Запроса** без изменения копируется в соответствующий пакет.

Если условия, необходимые для анализа не выполняются, сервер формирует и отсылает в ответ **Отказ** в доступе, при необходимости сопровождая его соответствующей текстовой информацией, которая будет отображена пользователю.

При соблюдении условий сервер может сразу аутентифицировать клиента пакетом **Подтверждение доступа** либо потребовать от него дополнительной информации, формируя в этом случае **Требование к доступу**, которое может содержать текстовую информацию для пользователей, необходимую для ответа на данный вопрос. Необходимо учесть, что RADIUS поддерживает аутентификацию пользователей по протоколам PAP и CHAP. Клиент, получив **Требование**, ожидает от пользователя ответ, затем, формируя новый **Запрос** (с новым идентификатором запроса), а атрибут пароля заменяет шифрованным **Ответом** пользователя и включает атрибут **Статуса** (если таковой присутствовал в **Требовании**).

Сервер отвечает на данный запрос либо пакетом **Подтверждение доступа**, либо пакетом **Отказ в доступе**, либо новым пакетом **Требование к доступу**. Если производится подтверждение доступа, то в пакет вносятся дополнительные атрибуты, необходимые для предоставления сервиса (например, IP-адрес, маска подсети и прочие возможные параметры).

Рассмотрим структуру пакета RADIUS (рис. 3.12).

Код	Идентификатор	Длина
Аутентификатор		
Атрибуты		

Рис. 3.12. Общая структура пакета RADIUS

Поле Код (1 октет) предназначено для указания типа пакета. В настоящее время определены следующие типы:

1 – **Запрос на доступ**.

2 – **Подтверждение доступа**.

3 – **Отказ в доступе**.
4 – **Запрос по учету**.
5 – **Ответ по учету**.
11 – **Требование к доступу**.
12 – **Статус Сервера**.
13 – **Статус Клиента**.
255 – **Зарезервировано**.

Использование типов 4 и 5 рассматривается в другом документе (RADIUS Accounting – [RFC-2866]). Типы 12 и 13 – экспериментальные.

Поле **Идентификатор** (1 октет) предназначено для обеспечения соответствия запрос-ответ.

Поле **Длина** (2 октета) указывает длину всего пакета, включая поля **Код**, **Идентификатор**, **Длина**, **Аутентификатор** и **Атрибуты**, и может указывать длину от 20 до 4096 байт. Если реально пакет меньше указанной длины, он не будет обработан. Все данные, превышающие указанную длину, не будут обработаны.

Поле **Аутентификатор** (16 октетов) имеет различные назначения. Для запросов это **Аутентификатор запроса** – случайное, уникальное, непредсказуемое число, обеспечивающее защиту от повторного использования перехваченных пакетов. Это значение в дальнейшем используется в качестве начального вектора режима шифрования CFB и указывается в атрибутах. Для пакетов **Подтверждения** или **Отказа в доступе** и **Требования к доступу** это **Аутентификатор ответа**, он представляет собой следующее значение MD5 (**Код** + **Идентификатор** + **Длина** + **Аутентификатор запроса** + **Атрибуты** + **Общий секрет**) – знак «+» означает конкатенацию («склеивание») соответствующих блоков данных.

Поле **Атрибуты переменной длины** представляет собой набор произвольного количества пар «Тип/Значение», которые содержат дополнительные данные, требующиеся для различных сервисов и прочих целей; его формат представлен на рис. 3.13.

Тип	Длина	Значение

Рис. 3.13. Формат поля **Атрибуты**

Поле **Тип** (1 октет) используется для указания типа атрибута. Значения этого поля не должны использовать следующие зарезервированные номера: 192–223 (экспериментальные), 224–240 (специальные), 241–255 (резервные). Из оставшихся в документе закреплены номера с 1 по 63. Не перечисляя все, укажем некоторые:

Имя пользователя – 1.
Пароль пользователя (в открытом виде) – 2.
Пароль пользователя (схема аутентификации CHAP) – 2.
IP-адрес NAS – 4.

Тип сервиса – 6.

Текстовое сообщение – краткий ответ пользователю – 18.

Двоичные параметры сессии в нерегламентированном формате (для частного использования) – 26.

Максимальная продолжительность сессии в секундах – 27.

Максимальное возможное бездействие в пределах сессии в секундах – 28.

Телефонный номер, набранный пользователем (в случае если он известен), – 30.

Телефонный номер пользователя (в случае если он определен) – 31.

Статус посредника – служебная информация, добавляемая RADIUS-сервером в тех случаях, когда он играет роль клиента другого RADIUS-сервера – 33.

Двоичные данные CHAP-запроса в тех случаях, когда сервер по каким-либо причинам не может разместить их в поле **Аутентификатор** – 60.

Поле **Длина** (1 октет) указывает длину записи об атрибуте, включая **Тип**, **Длину** и **Значение**.

Поле **Значение** переменной длины содержит атрибуты и может быть одним из пяти типов: *text* (текст), *string* (двоичные данные переменной **Длина**), *address* (32-битное целое – адрес), *integer* (32-битное целое) и *time* (32-битное целое – время в формате UNIX).

Рассмотрим документ [RFC-2866], регламентирующий учет в рамках протокола RADIUS.

Если клиент использует возможность учета, то при начале использования сервиса он формирует пакет **Начало учета** (Accounting start), указывающий тип сервиса и имя пользователя, использующего сервис, и отправляет пакет серверу учета, который подтверждает получение пакета. По окончании работы с сервисом клиент формирует и отправляет серверу пакет **Конец учета** (Accounting stop), в котором указывает тип сервиса и, возможно, статистика (время использования, объем полученных и отправленных данных и т.п.), сервер подтверждает получение.

Если сервер не присылает подтверждение, клиент совершает повторные попытки установленное число раз, а затем направляет данные альтернативному серверу. Важно отметить, что сервис учета является опциональным и никак не влияет на характер самой сессии. Если, например, NAS так и не получит от группы RADIUS-серверов подтверждение пакета **Начало учета**, то он все равно допустит пользователя к ресурсу. Более того, сначала происходит само подключение (т.е. сразу по приходу ответа **Подтверждение доступа**), а уже затем производится отправка пакетов учета. Для обеспечения учета в пакетах используются поле **Тип** с номерами с 40 по 51.

3.6 Протокол Kerberos

Протокол предназначен для аутентификации субъекта объектом (и наоборот), например, сервера – клиентом, в случае, когда среда передачи данных открыта, а объект изначально не знает о субъекте и не имеет с ним общего секрета, но оба (субъект и объект) предварительно идентифицированы третьей стороной – доверенным сервером – и имеют с ним общие секреты (никогда не передаваемые по сети). Требование наличия такого секрета и определяет схему защиты протокола – симметричными криптографическими алгоритмами (например DES, AES). Согласно принятой терминологии доверенный сервер называют центром распределения ключей – ЦРК. Рассмотрим наиболее известную реализацию протокола Kerberos в сетях на основе операционной системы Windows. В сетях с операционной средой Windows существует два типа аутентификации: интерактивная и неинтерактивная. Интерактивная сетевая аутентификация предполагает использование двух систем: системы клиента, с которой пользователь регистрируется, и контроллера домена, на котором хранится информация, относящаяся к паролю пользователя. Неинтерактивная аутентификация, которая может потребоваться для обеспечения уже зарегистрированному пользователю доступа к защищенным ресурсам, например, на сервере приложений, обычно использует три системы: клиента, сервера и контроллера домена, который выполняет необходимые для аутентификации операции от имени сервера.

Схема взаимодействия компонентов, используемых в процессах аутентификации в сетях с операционной средой Windows, приведена на рис. 3.14.

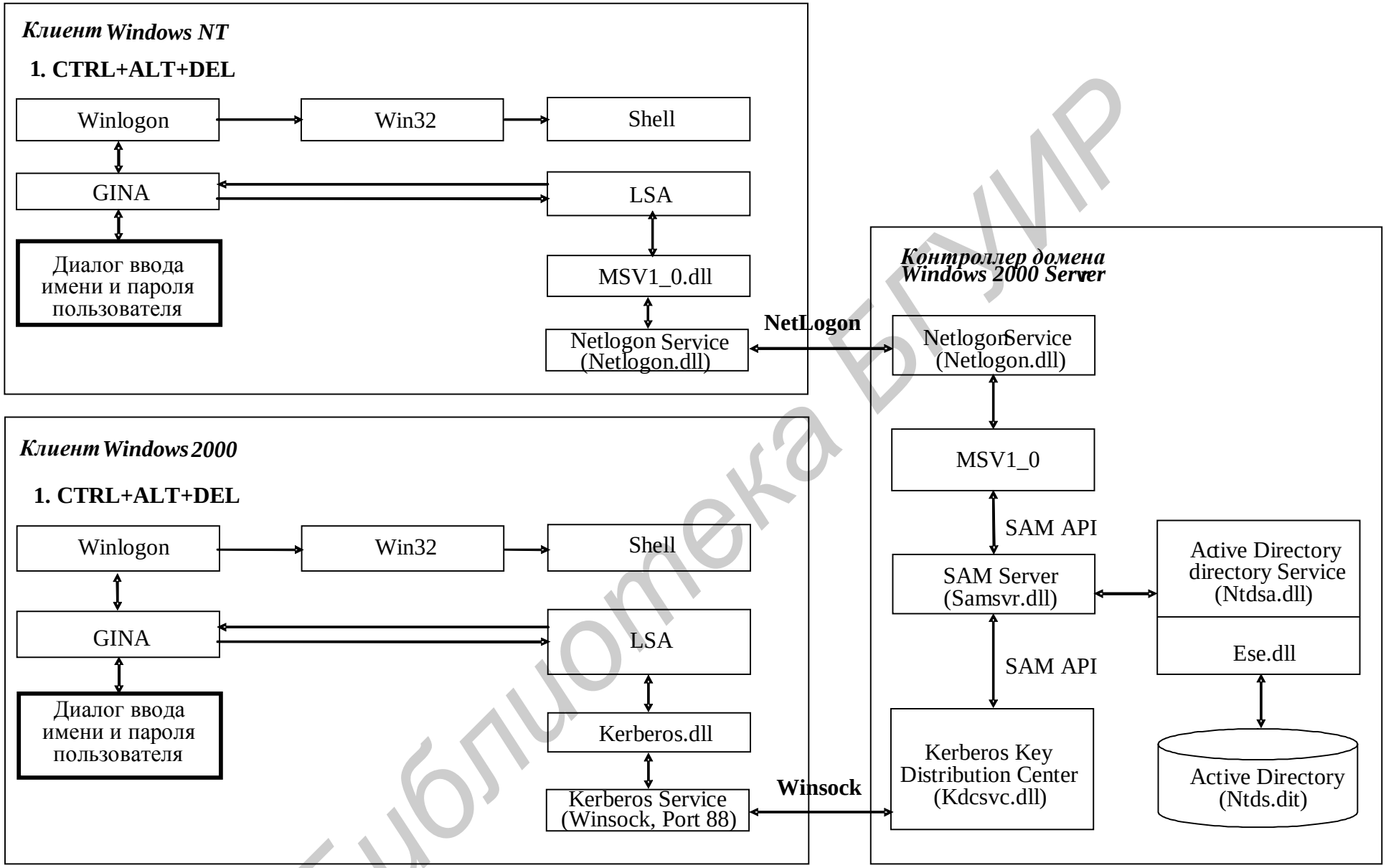

Рис. 3.14. Схема взаимодействия средств идентификации и аутентификации

Общий формат билета следующий.

1. Поле **Tkt–VNo** указывает номер версии протокола.

2. Поле **Realt** указывает имя области, где создан билет, т.е. доверенного ЦРК.

3. Поле **SName** указывает имя сервера, для которого предназначен билет.

4. Поле **Flage** – флаги билета, характеризующие его определённые свойства.

5. Поле **Key** – сессионный ключ шифрования.

6. Поле **CRealt** – наименование области, где зарегистрирован клиент и происходит первичная аутентификация.

7. Поле **CName** – идентификатор самого клиента.

8. Поле **Transited** – наименование областей, которые принимали участие в аутентификации клиента (не в порядке последовательности аутентификации).

9. Поле **AuthTime** – указывает время первоначальной аутентификации клиента из самого первого билета текущей сессии работы клиента (пользователя).

10. Поле **StartTime** – время вступления билета в силу (если не указано, берется равным предыдущему полю).

11. Поле **EndTime** – время окончания срока действия билета. Вместе с предыдущим полем составляет срок жизни билета. Некоторые службы устанавливают свои ограничения по времени и могут отвергнуть билет, который фактически не достиг срока истечения.

12. Поле **ReNew-Till** – определяет максимальное время, до которого билет может быть обновлен.

13. Поле **CAddr** – служит для указания адресов, с которых клиент может использовать данный билет. Если поле пустое – билет можно использовать с любого адреса.

14. Поле **Authorization-Data** – данные, которые могут быть использованы системой, получившей билет для управления полномочиями клиента.

Первые три поля из вышеуказанного списка – открытые, остальные поля шифруются.

В процессе функционирования системы Kerberos осуществляются несколько видов обмена сообщениями в зависимости от требуемых функций.

А. Обмен с сервисом начальной аутентификации (рис. 3.15).

Общение с сервисом начальной аутентификации обычно инициируется клиентом, желающим получить удостоверение к определенному серверу, но не имеющим пока других удостоверений. Для шифрования и расшифрования сообщений используется секретный ключ клиента. Как правило, данный обмен происходит при входе в систему для получения удостоверения к сервису выдачи билетов. Кроме того, общение с сервисом начальной аутентификации используется для получения удостоверения к серверам, требующим знания именно секретного ключа (пример – сервер смены пароля).

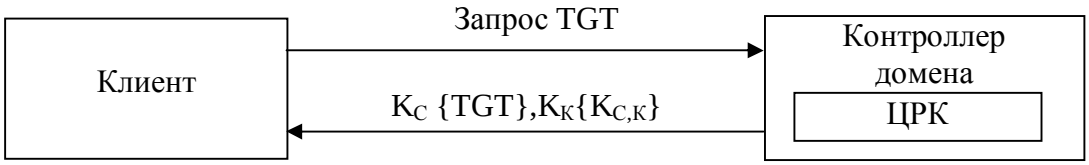

Рис. 3.15. Схема взаимодействия при начальной аутентификации

В своем запросе (KRB_AS_REQ) клиент посылает открытым текстом свое имя и имя сервиса, к которому он хочет получить удостоверение. Ответ (KRB_AS_REP) содержит билет (Ticket Granting Ticket – TGT), который клиент должен будет предоставить сервису выдачи билетов, и сеансовый ключ $K_{C,K}$ для совместного использования клиентом и ЦРК. Билет шифруется – секретным ключом ЦРК K_C, сеансовый ключ и дополнительная информация – секретным ключом клиента K_K. Сообщение KRB_AS_ REP содержит данные, позволяющие связать его с предыдущим запросом (KRB_AS_REQ) и обнаружить дублирование сообщений. В случае какой-либо ошибки возвращается сообщение KRB_ERROR, которое не шифруется.

Сервис аутентификации не делает попыток убедиться в подлинности обратившегося к нему субъекта. Он просто возвращает информацию, воспользоваться которой может лишь тот, кто знает секретный ключ субъекта.

Б. Обмен с сервисом выдачи билетов (TGS).

Обмен между клиентом и сервисом выдачи билетов инициируется клиентом, когда тот хочет получить удостоверение к определенному серверу или компьютеру (возможно, зарегистрированному в удаленной области управления), обновить или зарегистрировать существующий билет. Клиент должен располагать предварительно полученным от сервиса начальной аутентификации билетом к TGS. Формат сообщений при обмене с сервисом выдачи билетов почти тот же, что и для сервиса начальной аутентификации. Основное отличие состоит в том, что для шифрования и расшифрования используется сеансовый ключ.

Запрос (KRB_TGS_REQ) состоит из информации, подтверждающей подлинность клиента (TGT), и заявки на удостоверение. Ответ (KRB_TGS_REP) содержит запрашиваемое удостоверение, зашифрованное сеансовым ключом, и данные, позволяющие обнаружить дублирование сообщений и связать ответ с запросом.

В. Аутентификационный обмен «клиент-сервер».

Данный обмен используется сетевыми приложениями для взаимной проверки подлинности. Клиент должен располагать предварительно полученным удостоверением к серверу. Он передает серверу билет, аутентификатор (зашифрованный сеансовым ключом) и некоторую дополнительную учетную информацию. Сервер в ответ возвращает только аутентификатор, зашифрованный тем же сеансовым ключом.

Чтобы с помощью Kerberos получить доступ к серверу S (рис. 3.16), клиент С посылает системе Kerberos запрос, содержащий сведения о нем (клиенте) и о

запрашиваемой услуге. В ответ Kerberos возвращает информацию двух видов: билет, зашифрованный секретным ключом сервера K_S, и копию части информации из билета, зашифрованную секретным ключом клиента. Клиент должен расшифровать вторую часть данных и переслать ее вместе с билетом серверу. Сервер, расшифровав билет, может сравнить его содержимое с дополнительной информацией, присланной клиентом. Совпадение свидетельствует о том, что клиент смог расшифровать предназначенные ему данные (ведь содержимое билета никому, кроме сервера и Kerberos, недоступно), т.е. продемонстрировал знание своего секретного ключа. Значит, клиент – именно тот, за кого себя выдает.

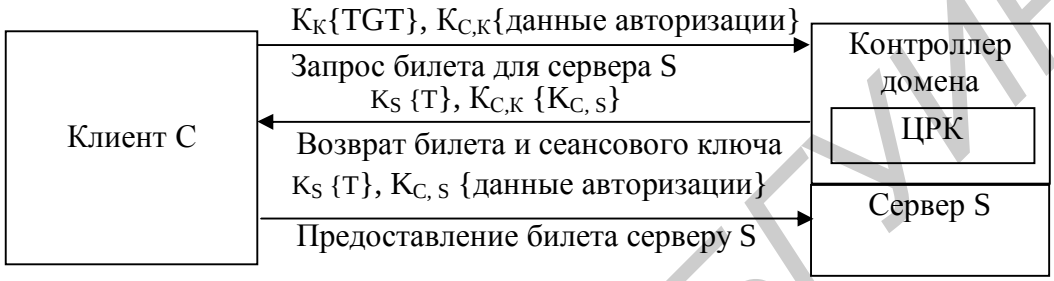

Рис. 3.16. Схема аутентификационного обмена «клиент – сервер»

Следует подчеркнуть, что секретные ключи в процессе проверки подлинности не передаются по сети (даже в зашифрованном виде) – они только используются для шифрования.

Контрольные вопросы и задачи

1. В чем отличие протоколов аутентификации PAP и CHAP?
2. Разработать алгоритм реализации протокола RADIUS.
3. Пояснить сущность и особенности протоколов аутентификации в соединениях «точка-точка».
4. Как определяется подлинность пользователя при обращении к сервисам в сетях с операционной средой Windows 2000?
5. Что такое одноразовое случайное число? Что такое метка времени? Какую роль они играют в протоколах аутентификации и протоколах ключа взаимосвязи?
6. Какие действия может предпринимать активный нарушитель?
7. Почему, несмотря на идеальное шифрование и идеальную аутентификацию сообщений, протоколы аутентификации остаются уязвимыми?
8. Почему в протоколе Kerberos каждому клиенту должно соответствовать три разных вида серверов?
9. Чем протокол Kerberos удобен для применения в корпоративных сетях?

ГЛАВА 4. ПРОТОКОЛЫ АУТЕНТИФИКАЦИИ В ИНТЕРНЕТ

Интернет представляет собой огромную открытую сеть компьютеров, называемых узлами. Каждый узел имеет уникальный сетевой адрес, так что сообщение, посланное на этот узел или отосланное с этого узла, содержит его сетевой адрес. Протокол, обеспечивающий передачу сообщений в сети, называется протоколом Интернет (Internet Protocol – IP). По этой причине уникальный сетевой адрес называется IP-адресом узла. В соответствии с эталонной семиуровневой моделью взаимосвязи открытых систем протокол IP работает на третьем уровне. Этот уровень называется сетевым или IP-уровнем. Связь на IP-уровне осуществляется с помощью IP-пакетов. На рис. 4.1 изображен IP-пакет, не имеющий криптографической защиты.

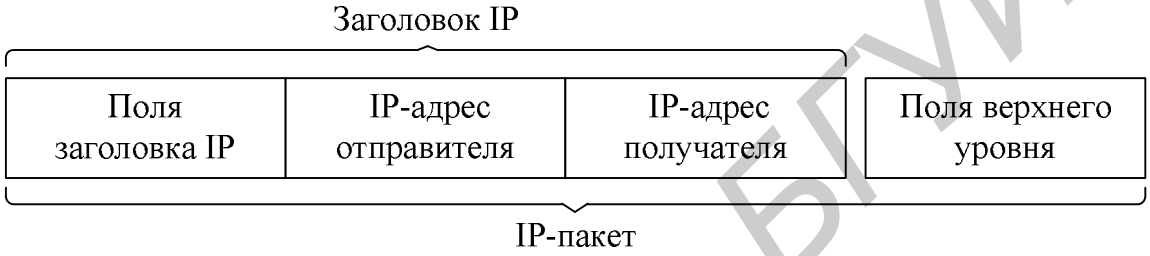

Рис. 4.1. Незащищенный IP-пакет

Назначение первых трех полей IP-пакета ясно из названий. Четвертое поле «Поля верхнего уровня» содержит, во-первых, спецификацию протокола, запускаемого на ближайшем верхнем уровне и обрабатывающего IP-пакет (т.е. «протокола управления передачей» – (Transmision Control Protocol – TCP) и, во-вторых, данные, содержащиеся в IP-пакете.

Стороны, обменивающиеся сообщениями, могут установить секретную связь, применив оперативное шифрование, используя общий или открытый ключ. Поскольку оперативное шифрование осуществляется на уровне приложений, зашифровывается только четвертое поле IP-пакета. Если протокол Интернет, применяемый пользователями, не обеспечивает безопасности, поля данных в IP-заголовке не защищаются. Изменение данных, содержащихся в этих полях, открывает много возможностей для организации разнообразных атак, связанных с нарушением безопасности сети.

4.1. Протокол обеспечения безопасности в Интернет

Протокол обеспечения безопасности в Интернет, известный под названием IPSec, предназначен для криптографической защиты IP-заголовка, состоящего из трех полей IP-пакета (рис. 4.1). Этот протокол предусматривает обязательную аутентификацию IP-заголовка и возможную защиту информации об отправителе и адресате, содержащейся в некоторых полях IP-заголовка. Спецификации IPSec определяются целым рядом документов. Наиболее важными из них являются следующие:

RFC 2401 – архитектура безопасности для IP;

RFC 2402 – описание расширений аутентификации пакетов IP;

RFC 2406 – описание расширений шифрования пакетов IP;

RFC 2408 – спецификации средств управления ключами.

В соответствии с указанными документами средства защиты реализуются в виде заголовков расширений, которые следуют за основным заголовком IP. Заголовок расширения аутентификации называют заголовком AH (Authentication Header – заголовок аутентификации), а заголовок расширения шифрования – заголовком ESP (Encapsulating Security Payload header – заголовок защищённого полезного груза или заголовок защищённого содержимого).

В дополнение к этим четырём документам протокол IPSec включает ещё семь групп документов (рис. 4.2).

Архитектура содержит описание общих принципов, требований защиты, а также определения и механизмы реализации технологии IPSec.

Безопасное сокрытие значимых данных (ESP) содержит описание формата пакета и общих принципов использования ESP для шифрования и аутентификации пакетов.

Заголовок аутентификации (AH) включает описание формата пакета и общих принципов использования AH для аутентификации пакетов.

Алгоритм шифрования представляет собой набор документов, определяющих использование различных алгоритмов шифрования для ESP.

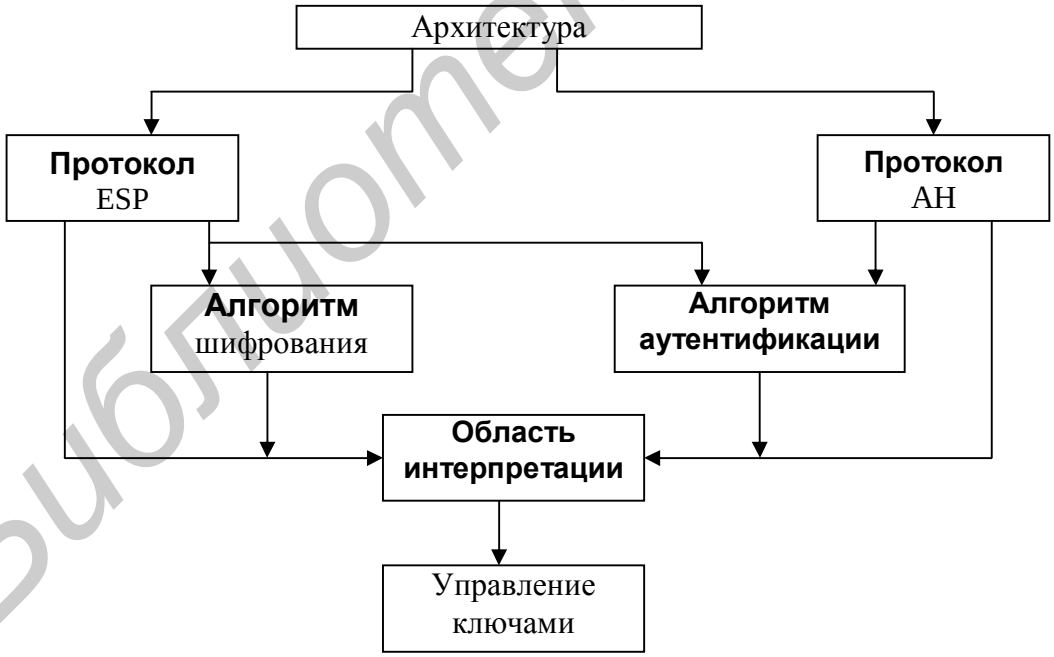

Рис. 4.2. Общая структура документов IPSec

Алгоритм аутентификации – набор документов, определяющих использование различных алгоритмов аутентификации для AH и опции аутентификации ESP.

Область интерпретации содержит значения, необходимые для соответствия одних документов другим – в частности, идентификаторы проверенных алгоритмов шифрования и аутентификации, а также некоторые параметры, например, продолжительность жизненного цикла ключей.

Управление ключами – документы, описывающие схемы управления ключами.

IPSec обеспечивает сервис защиты на уровне IP, позволяя системе выбрать необходимые протоколы защиты, определить алгоритм для соответствующего сервиса и задать значения любых криптографических ключей, требующихся для запрашиваемого сервиса. Для защиты используется два протокола: протокол аутентификации, указанный заголовком аутентификации AH, и комбинированный протокол шифрования/аутентификации, определяемый форматом пакета для протокола ESP. В табл. 4.1 показаны сервисы, обеспечивающие применение указанных протоколов.

Ключевым элементом реализации указанных сервисов является защищённая связь SA (security association). Связь представляет собой одностороннее отношение между отправителем и получателем, применяющим сервис защиты к транспортному потоку. Если требуется равноправное отношение для двухстороннего защищённого обмена, необходимы две защищённые связи. Сервис защиты даёт возможность для защищённой связи использовать либо AH, либо ESP, но никак не оба протокола одновременно.

Таблица 4.1

Сервисы, обеспечивающие применение протоколов AH, ESP

Наименование сервиса	Протокол AH	Протокол ESP (только шифрование)	Протокол ESP (шифрование и аутентификация)
Управление доступом	×	×	×
Целостность без установки соединений	×		×
Аутентификация источника данных	×		×
Защита от воспроизведения пакетов	×	×	×
Конфиденциальность		×	×
Конфиденциальность потока		×	×

Защищённая связь однозначно определяется следующими тремя параметрами.

1. **Индекс параметров защиты** – строка битов, присваиваемая конкретной защищённой связи. Индекс параметров защиты передаётся в заголовках AH и ESP, чтобы принимающая система имела возможность выбрать защищённую связь, в рамках которой должен обрабатываться принимаемый пакет.

2. **Адрес IP пункта назначения** – адрес пункта назначения защищённой связи, который может представлять систему конечного пользователя или сетевой объект тапа VPN-агента.

3. **Идентификатор протокола защиты** указывает, является ли данная защищённая связь защищённой связью АН или это защищённая связь ESP.

Механизм управления ключами связывается с механизмами аутентификации и конфиденциальности только через параметры защиты. Таким образом, он может быть определён независимо от механизмов аутентификации и конфиденциальности.

4.1.1. Протокол АН

Заголовок аутентификации (АН) обеспечивает поддержку целостности данных и аутентификацию пакетов IP. Функция аутентификации позволяет VPN-агенту идентифицировать пользователя или приложение, а также защититься от очень распространённых сегодня в Интернет атак с подменой сетевых адресов и несанкционированного воспроизведения сообщений. Аутентификация опирается на использование кодов аутентичности сообщений, при этом две стороны должны использовать общий секретный ключ.

Заголовок аутентификации состоит из следующих полей (рис. 4.3).

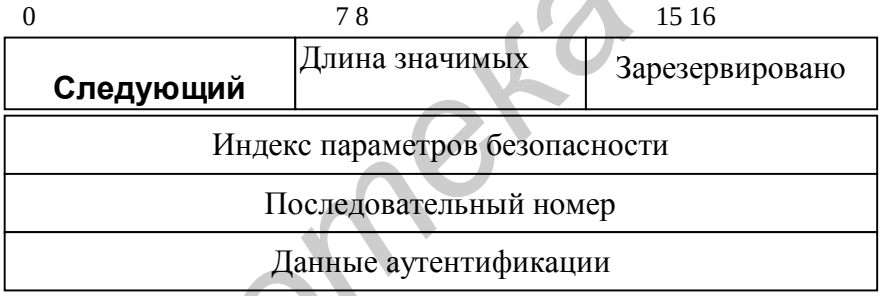

Рис. 4.3. Формат АН

Следующий заголовок (8 бит). Идентифицирует тип заголовка, следующего непосредственно за данным заголовком.

Длина значимых данных (8 бит). Длина заголовка аутентификации в 32-битовых словах.

Зарезервировано (16 бит). Значение поля должно быть нулевым.

Индекс параметров защиты (32 бит). Идентифицирует защищённую связь.

Последовательный номер (32 бит). Монотонно возрастающий номер в диапазоне от 0 до $2^{32} - 1$, использующийся для нумерации пакетов.

Данные аутентификации (переменной длины). Поле переменной длины, содержащее код аутентификации сообщения для данного пакета.

Когда устанавливается новая защищённая связь, отправитель инициализирует счётчик последовательных номеров, установив соответствующее значение равным 0. Каждый раз, когда по защищённой связи посылается пакет, отправитель

увеличивает значение данного счётчика и размещает его в поле последовательных номеров. На приёмной стороне осуществляется контроль последовательности номеров принимаемых пакетов, и пакеты с одинаковыми номерами отбрасываются. Когда значение счётчика превысит значение $2^{32} - 1$, отправитель должен завершить данную защищённую связь и инициализировать новую защищённую связь с новым ключом.

Для вычисления кода аутентификации сообщения выбирается следующая информация:

– поля заголовка IP, которые либо не изменяются в пути следования (неизменяемые поля), либо имеют прогнозируемые значения в пункте назначения защищённой связи. Поля, которые могут измениться в пути следования, и значения которых в конечной точке нельзя предсказать, обнуляются для вычислений в пункте отправления и пункте назначения;

– заголовок АН, за исключением поля данных аутентификации, которое обнуляется для вычислений в пунктах отправления и назначения;

– все данные протокола вышеследующего уровня (т.е. сам внутренний пакет IP), которые должны оставаться неизменными в пути следования.

Имеющиеся на сегодня спецификации протокола требуют, чтобы любая реализация поддерживала следующие алгоритмы для вычисления кода аутентификации сообщения: HMAC – MD5 и HMAC – SHA-1. Кроме того, протокол АН может быть реализован с использованием отечественных алгоритмов хеширования и формирования цифровой подписи в соответствии с СТБ 1176.1-99 и СТБ 1176.2-99, а также российских алгоритмов хеширования и формирования цифровой подписи в соответствии с ГОСТ Р 34.11-94 и ГОСТ Р 34.10-2001 или алгоритма криптографического преобразования по ГОСТ 28147-89 в режиме вычисления имитовставки.

4.1.2. Протокол ESP

Отличительной функцией протокола ESP является обеспечение конфиденциальности путём шифрования внутреннего пакета IP. Пакет ESP содержит следующие поля (рис. 4.3).

Индекс параметров защиты (32 бит). Идентифицирует защищённую связь.

Последовательный номер (32 бит). Значение счётчика, используемого для защиты от атак воспроизведения, как и при использовании протокола АН.

Значимые данные (переменной длины). Внутренний пакет IP, который защищается шифрованием.

Заполнитель (0 – 255 байт). Поле заполняется нулями до кратности целому числу байтов в соответствии с форматами блоков используемых алгоритмов шифрования.

Длина заполнителя (8 бит). Указывает число байт заполнителя, предшествующего данному полю.

Следующий заголовок (8 бит). Идентифицирует тип данных, содержащихся в поле значимых данных, указывая первый заголовок значимых данных.

Данные аутентификации (переменной длины). Поле переменной длины, содержащее код аутентификации сообщения, вычисляемый для данного пакета ESP, без поля данных аутентификации.

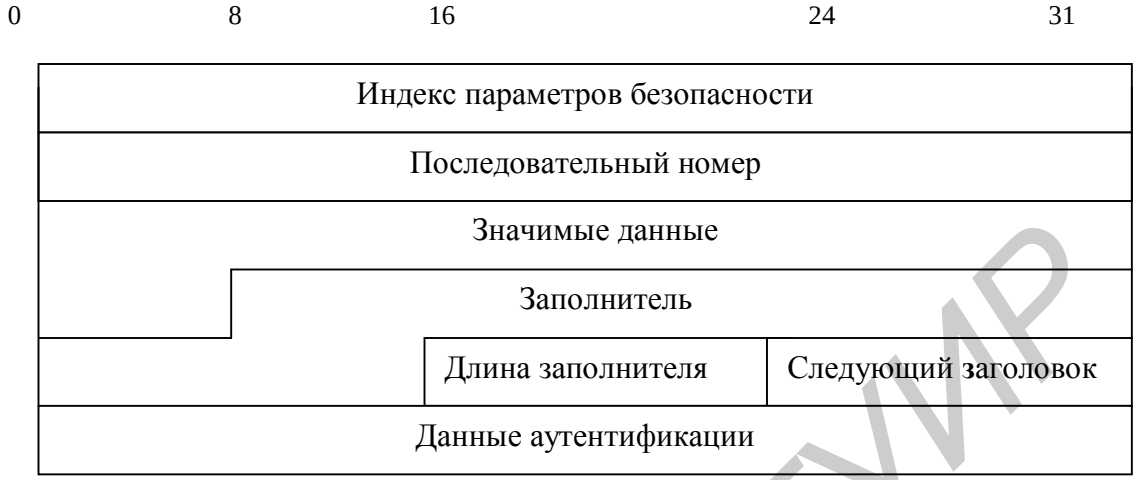

Рис. 4.4. Формат ESP

Сервис ESP предполагает шифрование полей значимых данных, заполнителя, длины заполнителя и следующего заголовка. Если для алгоритма, используемого при шифровании значимых данных, требуется синхронизация данных, то необходимая синхропосылка вставляется в начало поля значимых данных.

Существующие спецификации требуют, чтобы любая реализация поддерживала использование алгоритма DES в режиме со сцеплением блоков (CBC), однако могут использоваться и другие алгоритмы шифрования, например 3DES, RC5, AES и др. При обеспечении аутентификации сообщений в протоколе ESP используются механизмы, рассмотренные в гл. 3.

4.1.3. Режимы работы протокола

Каждый из протоколов AH и ESP поддерживает два режима работы – транспортный и туннельный.

В *транспортном* режиме механизмы безопасности применяются только для протоколов, начиная с транспортного (TPC) уровня и выше, оставляя данные самого сетевого уровня (заголовок IP) без дополнительной информации, вставляемой протоколами в пакет (рис. 4.5).

Туннельный режим характеризуется тем, что обеспечивает защиту также и данных сетевого уровня путем добавления нового IP-заголовка. После определения ассоциаций безопасности (например, между двумя шлюзами) истинные адреса хостов отправления и назначения (и другие служебные поля) полностью защищаются от модификаций для AH или вообще скрываются для ESP, а в новый заголовок выставляются адреса и другие данные шлюзов (отправления/получения). На рис. 4.6 видны преимущества и недостатки обоих протоколов. Так, ESP обеспечивает сокрытие данных, но не скрывает данные. По этой причине для обеспечения высокого уровня безопасности применяется совмещение протоколов.

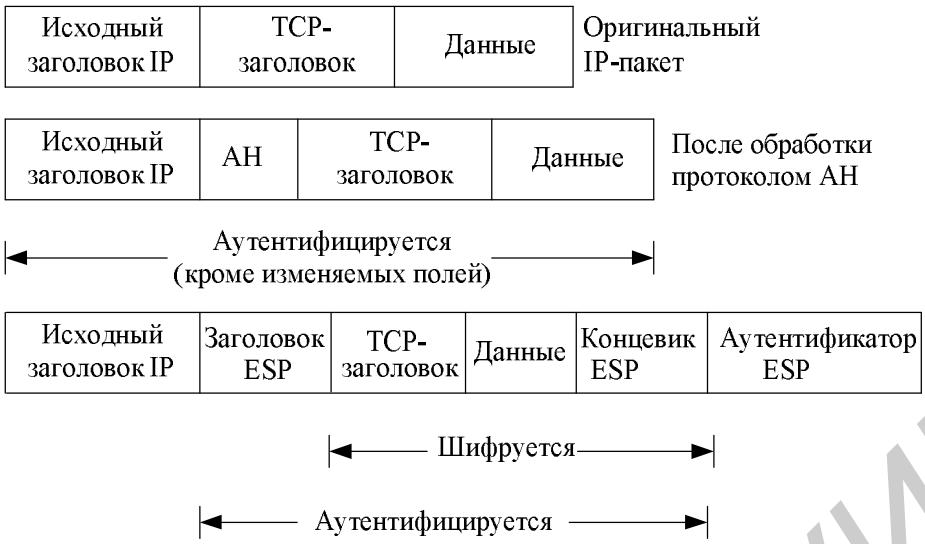

Рис. 4.5. Транспортный режим

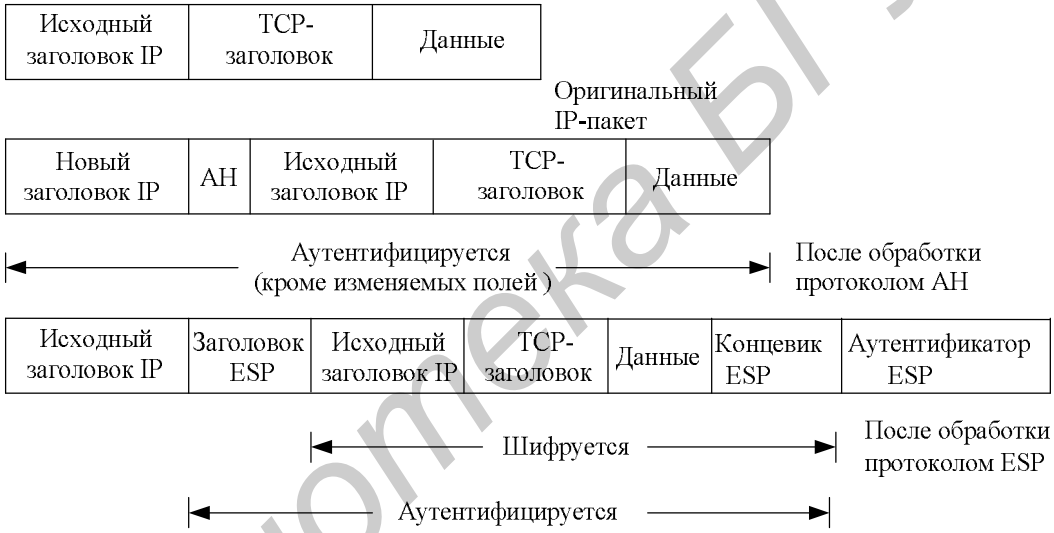

Рис. 4.6. Туннельный режим

4.1.4. Ассоциация безопасности

Концепция Ассоциаций безопасности (SA) является основополагающей для IPSec. SA определяет взаимоотношения между двумя или более участниками безопасной связи и описывает, какие сервисы будут использованы для обеспечения безопасности: алгоритмы шифрования, алгоритмы аутентификаций и общие сессионные ключи.

SA всегда однонаправлена, поэтому для одного двунаправленного соединения между двумя участниками требуется две SA (по одной на каждое направление).

Набор всех SA, установленных на узле для связи с другим узлом (узлами), хранится в специальной базе данных ассоциаций безопасности (SAD). Каждый узел поддерживает две SAD (одну для входящего трафика).

В зависимости от реализации может потребоваться несколько пар SAD для мультиинтерфейсных узлов – по одной SAD на каждый интерфейс.

Набор всех SA, установленных на узле для связи с другим узлом (узлами), хранится в специальной базе данных ассоциаций безопасности (Security Association Database – SAD). Каждый узел поддерживает две SAD; одна из них для входящего трафика. В зависимости от реализации может потребоваться несколько пар SAD для мультиинтерфейсных узлов – по одной SAD на каждый интерфейс.

Когда у узла существует несколько соединений с другим узлом (узлами), необходимо определить, которую из SA применять к пакетам какого соединения. Для этой цели служит база данных политик безопасности (Security Policy Database – SPD). Запись SPD состоит из полей, связывающих SA с идентификатором пакетов соединения – селектором. Селектор состоит из следующих параметров:

– IP-адрес назначения (индивидуальный, групповой, широковещательный), диапазон адресов + маска или групповой символ;

– IP-адрес источника (индивидуальный, групповой, широковещательный), диапазон адресов + маска или групповой символ.

– имя, в двух вариантах:

1) идентификатор пользователя в формате DNS или X.500;

2) наименование системы в формате DNS или X.500;

– уровень секретности данных по меткам IPSO/CIPSO (таким как «не определено», «коммерческая собственность», «секретно» и т.п.) – этот параметр опционален;

– протокол транспортного уровня;

– порт назначения;

– порт источника.

4.2. Протокол обмена ключами через Интернет

Включение в протокол IPSec заголовка аутентификации и инкапсулированной защиты обеспечивает криптографическую защиту IP-пакета. Однако два узла, устанавливающих между собой связь, сначала должны согласовать средства защиты (криптографические ключи, алгоритмы, параметры). Для этого предназначен протокола обмена ключами через Интернет (Internet Key Exchange – IKE). В настоящее время он является стандартным протоколом обмена ключами в рамках протокола IPSec, одобренным группой IETF.

Протокол IKE представляет собой набор протоколов аутентификации и обмена аутентифицированными ключами. Каждый протокол из этого набора представляет собой гибрид, использующий часть протокола Окли (протокола определения ключа Окли), часть механизма SKEME (механизм для многостороннего обмена секретными ключами через Интернет) и часть

протокола ISAKMP (протокол установления защищенных соединений и управления ключами).

Протокол определения ключей Окли описывает режимы обмена ключами и специфицирует предоставляемые ими функции.

Механизм безопасного обмена ключами в Интернете – SKEMI – описывает многофункциональные технологии, предоставляющие анонимность, неотрекаемость и быстрое обновление ключей.

Протокол ассоциаций безопасности и управления ключами в Интернете – ISAKMP – устанавливает общие инструкции для обеспечения аутентификации и обмена ключами без указания конкретных прикладных алгоритмов.

В ходе установления ассоциаций безопасности, IKE согласовывает следующие атрибуты: алгоритм шифрования, алгоритм хэширования, метод аутентификации и данные о группе преобразования алгоритма Диффи-Хеллмана.

Аутентификация может быть произведена с помощью следующих методов:

– предоставленного секрета (секретная информация, известная обеим сторонам соединения до начала установки соединения);

– криптографии с открытым ключом (обе стороны генерируют случайное число и шифруют его вместе со своим идентификатором открытым ключом противоположной стороны);

– электронной цифровой подписью (стороной используется закрытый ключ, открытый ключ которого есть у противоположной стороны).

Протокол IKE включает две фазы согласования ключей. В первой фазе происходит создание защищенного канала, во второй – согласование и обмен ключами, установление SA. Первая фаза использует один из двух режимов: основной или агрессивный. Различие между ними в уровне защищенности и скорости работы. Основной режим, более медленный, защищает всю информацию, передаваемую между узлами. Агрессивный режим для ускорения работы оставляет ряд параметров открытыми и уязвимыми для прослушивания и рекомендуется использовать только в случае, когда критическим вопросом является скорость работы. Во второй фазе используется быстрый режим, названный так потому, что он производит аутентификации узлов, считая, что это было сделано в первой фазе, обеспечивающей обмен ключами для шифрования данных.

4.3. Протокол защищенных сокетов

Для обеспечения безопасности в сети World Wide Web используется протокол защищенных сокетов SSL (Secure Sockets Layer Protocol). Термин «сокет» обозначает стандартный канал связи, соединяющий процессы, протекающие в сети (например, клиентов и сервера). Протокол защищенных сокетов выполняется в рамках протоколов на уровне приложений, например, в рамках протокола передачи гипертекстовых файлов HTTP (Hypertext Transfer Protocol), облегченного протокола службы каталогов LDAP (Lightweight

Directory Access Protocol) или протокола доступа к сообщениям в сети Интернет IMAP (Internet Messagging Access Protocol), а также поверх стека протоколов TCP/IP. Если соединения на уровне сокетов защищены (например, на предмет секретности и целостности данных), эта защита распространяется на все соединения в рамках протоколов на уровне приложений. Протокол SSL в настоящее время является одним из самых популярных протоколов безопасности транспортного уровня, используемых в Интернете. Работая на транспортном уровне стека TPC/IP, он решает следующие задачи:

– обеспечивает конфиденциальность данных, т.е. гарантирует, что они не были раскрыты в ходе транспортировки между клиентом и сервером путем шифрования данных всех вышестоящих уровней (представления и прикладного), при этом открытой остается только служебная информация уровня TCP и ниже;

– обеспечивает аутентификацию сервера, т.е. пользователь (клиент) может быть уверен, что он получает доступ именно к тому серверу, к которому необходимо;

– опционально может обеспечивать аутентификацию клиента, т.е. гарантировать серверу, что он работает с авторизованным клиентом;

– обеспечивает целостность передаваемой информации, т.е. гарантирует, что информация не была изменена в ходе транспортировки между клиентом и сервером;

– опционально может сжимать данные для обеспечения скорости передачи.

Протокол транспортного уровня TLS (Transport Layer Security) [RFC-2246] является преемником протокола SSL и стандарта безопасности в World Wide Web и разработан проблемной группой проектирования Internet (Internet Engineering Task Force – IETF). Протокол TLS основан на протоколе SSL, при этом их различия невелики, поэтому в дальнейшем при описании протокола защиты данных в Internet будем рассматривать протокол TLS. Он, в свою очередь, состоит из двух протоколов – протокола записи, определяющего формат передачи данных, и протокола взаимосвязи, определяющего механизм установки соединения.

4.3.1. Протокол записи

Протокол записи TLS обеспечивает безопасную инкапсуляцию канала связи для применения в рамках протокола приложений более высокого уровня. Протокол TLS запускается поверх протоколов TCP и IP и обеспечивает надежный сеанс связи. Он распределяет данные по блокам, сжимает их при необходимости, применяет код аутентификации сообщения для защиты целостности данных, шифрует сообщение с помощью симметричного алгоритма и передает результат адресату. Адресат получает зашифрованные блоки данных, расшифровывает их, верифицирует код MAC, разархивирует их при необходимости, собирает блоки в одно целое и доставляет результат на более высокий уровень приложений.

Ключи для симметричного шифрования и код MAC генерируются для каждого сеанса связи отдельно и основаны на секрете, согласованном в протоколе

взаимосвязи TLS. Кроме того, этот протокол берет на себя функции генерирования системных сообщений об ошибках и закрытии сессии. Рассмотрим возможные варианты служебных сообщений (табл. 4.2).

Таблица 4.2

Системные сообщения протокола записи

Сообщение	Значение сообщения
Close_notify	Это нормальное, не связанное с ошибкой, сообщение о закрытии сессии
Unexpected_message	Получено несоответствующее сообщение, которое приводит к закрытию сессии
Bad_record_mac	Получен неверный аутентификационный код сообщения, приводящий к закрытию сессии
Decompression_failure	Функция распаковки получила неверное значение; приводит к закрытию сессии
Handshake_failure	Получатель не способен поддержать требуемый отправителем набор параметров безопасности, что приводит к закрытию сессии
No_certificate	Нет доступа к сертификату
Bad_ certificate	Сертификат поврежден
Unsup-ported_certificate	Данный тип сертификата не поддерживается
Certificate_revoked	Сертификат отозван выпустившим его субъектом
Certificate_unknown	При обработке данных сертификата возникли прочие неразрешимые проблемы
Illegal_parameter	Поле данных имеет неверное значение, что приводит к закрытию сессии

4.3.2. Протокол взаимосвязи

Протокол взаимосвязи обеспечивает настройку множества криптографических параметров. В момент, когда клиент пытается установить соединение с сервером, обе стороны осуществляют следующие действия:

– обмениваются приветствиями, согласовывая алгоритм, пересылают друг другу случайные числа и проверяют возможность возобновления сеанса;

– обмениваются необходимыми криптографическими параметрами, позволяющими клиенту и серверу согласовать секрет (так называемый «главный секрет»);

– обмениваются сертификатами и криптографической информацией, позволяющей клиенту и серверу аутентифицировать друг друга;

– генерируют сеансовые секреты на основе главного секрета, обмениваясь случайными числами;

– убеждаются, что их партнер вычислил те же самые параметры безопасности, протокол выполнен успешно и не подвергся атаке;

– установленный защищенный канал передается протоколу записи TSL для обработки приложений на более высоком уровне.

Перечисленные действия реализуются с помощью четырех этапов-обменов, описанных ниже.

4.3.2.1. Обмен приветствиями

Сеанс начинается с того момента, когда клиент посылает сообщение ClientHello, на которое сервер должен ответить сообщением ServerHello. В противном случае связь прерывается. Эти два сообщения заполняют следующие поля: protocol_version («версия_протокола»), random («случайные_числа»), session_id («номер_сеанса»), cipher_suites («элементы_шифра») и compression_message («методы_сжатия»).

Поле **protocol_version** обладает свойством обратной совместимости: сервер и клиент могут использовать это поле для того, чтобы информировать своего партнера о версии используемого протокола.

Поле **random** содержит одноразовые случайные числа (идентификаторы «актуальности»). Партнеры генерируют эти числа для последующего обмена. Кроме того, данное поле содержит локальное время установления связи каждой из сторон.

Поле **session_id** идентифицирует текущее сеансовое соединение. В начале нового сеанса связи поле **ClientHello.session_id** должно оставаться пустым. В этом случае сервер генерирует новый номер сеанса, записывает его в поле **ServerHello.session_id** и сохраняет в локальной кэш-памяти. Если поле **ClientHello.session_id** содержит какое-то число (например, когда клиент желает возобновить текущий сеанс), сервер должен попытаться найти номер сеанса в локальной кэш-памяти и возобновить указанный сеанс.

Поле **ClientHello.cipher_suites** представляет собой список криптографических опций, поддерживаемых на клиентской машине и упорядоченных в соответствии с приоритетами клиента. Клиент может предложить серверу широкий круг криптографических алгоритмов (как симметричных, так и с открытым ключом), кодов аутентификации сообщений и функций хэширования. Для каждой криптографической операции сервер выбирает единственную схему и сообщает о ней клиенту, используя поле **ClientHello.cipher_suites.**

4.3.2.2. Предложения ключей сервером

После обмена приветствиями, если сервер должен аутентифицировать себя, он высылает клиенту свой сертификат. Если сообщение ServerCertificate не является пустым, оно содержит список сертификатов X.509.v3. Сертификат X.509 содержит информацию об имени и открытом ключе владельца сертификата, а также об источнике сертификата. Получив список сертификатов, клиент может выбрать алгоритм с открытым ключом, поддерживаемый клиентской машиной.

В случае если у него нет соответствующего сертификата, он высылает сообщение Server_key_exchange. Оно содержит элементы открытых ключей, соответствующих списку сертификата в сообщения ServerCertificate. Элементы ключа Диффи-Хеллмана включаются в тройку (p, g, g^y), где p – простой модуль, g – порождающий элемент большой группы по модулю p, а y – целое число, записанное в кэш-памяти сервера и связанное с полем «номер_сеанса».

Если требуется аутентификация клиента, сервер может послать сообщение Certificate_request, запрашивая сертификат клиента. После этого сервер отправляет сообщение Server_hello_done, показывая, что он завершил свою работу и ожидает сообщений от клиента.

4.3.2.3. Ответ клиента

Получив сообщение CertificateRequest, клиент должен в ответ отослать либо сообщение ClientCertificate, либо предупреждение NoCertificate.

Затем клиент пересылает серверу сообщение ClientKeyExchange. Содержание этого сообщения зависит от выбранного алгоритма с открытым ключом, согласованным путем обмена сообщениями ClientHello и ServerHello.

После этого клиент генерирует «главный секрет» и шифрует его с помощью сертифицированного открытого ключа, принадлежащего серверу и содержащегося в сообщении ServerCertincate.

Если клиент использовал сертификат для своей аутентификации, то он отсылает серверу сообщение CertincateVerify, снабженное цифровой подписью. Это позволяет серверу явно верифицировать сертификат клиента.

4.3.2.4. Обмен заключительными сообщениями

На этом этапе клиент формирует сообщение Change_cipher_spec, которое означает, что клиент произвел выбор из списка рассматриваемых параметров шифрования и перенес их в статус текущих. Затем клиент отправляет сообщение ClientFinished, означающее, что этап согласования закончен, при этом данное сообщение уже зашифровано в рамках новых параметров шифрования. Сервер также производит установку текущих параметров шифрования и отвечает сообщениями Change_cipher_spec и ServerFinished, позволяющими клиенту убедиться в согласовании параметров, выполненном сервером.

Таким образом, взаимосвязь считается установленной, клиент и сервер могут переходить к обмену данными на уровне приложений.

4.4. Протокол удалённой регистрации

Протокол удалённой регистрации SSH (Secure Shell) – это набор протоколов аутентификации с открытым ключом, позволяющий пользователю, работающему на клиентской машине, безопасно регистрироваться на удаленном сервере через ненадежную сеть, выполнять команды на удаленном сервере и перемещать файлы с одного компьютера на другой. Основная идея протокола SSH состоит в том, что пользователь, работающий на клиентской машине, должен загрузить с удаленного сервера открытый ключ и установить с его помощью защищенный канал, используя криптографический мандат. В случае, если криптографическим мандатом пользователя является его пароль, он шифруется с помощью полученного открытого ключа и передаётся на сервер. Этот протокол фактически стал промышленным стандартом и состоит из трех нижеперечисленных протоколов.

1. Протокол транспортного уровня (SSH-TRANS), обеспечивающий аутентификацию сервера, конфиденциальность и целостность данных. Этот протокол использует открытый ключ. Исходной информацией для этого протокола как со стороны сервера, так и со стороны клиента является пара открытых ключей, называемая «ключом головного компьютера» («host key»). Результатом протокола является взаимно аутентифицированный защищенный канал, гарантирующий секретность и целостность данных, направленных от сервера к клиенту. Этот протокол обычно выполняется на основе протоколов TCP/IP, однако может использоваться поверх любых других надежных протоколов передачи данных.

2. Протокол аутентификации пользователя (SSH-USERAUTH) аутентифицирует клиента перед сервером. Этот протокол выполняется по каналу односторонней аутентификации, установленному протоколом транспортного уровня SSH. Он поддерживает работу различных протоколов односторонней аутентификации для того, чтобы выполнить аутентификацию в направлении от клиента к серверу. Для того чтобы аутентификация в этом направлении стала возможной, удаленный сервер должен иметь априорную информацию о криптографическом мандате пользователя, т.е. пользователь должен быть известен серверу. Протоколы данного типа могут применять либо открытый ключ, либо пароль. Например, они могут быть созданы на основе протокола аутентификации с помощью простого пароля. Результатом выполнения протокола аутентификации пользователя SSH является взаимно аутентифицированный защищенный канал между сервером и пользователем, который функционирует поверх протокола транспортного уровня.

3. Протокол соединения (SSH-CONN) совмещает различные логические каналы в единый шифрованный канал. Этот протокол выполняется по взаимно аутентифицированному защищенному каналу, установленному двумя предыдущими протоколами. Он обеспечивает работу защищенного канала и разделяет его на несколько защищенных логических каналов, используя стандартные методы организации интерактивных сеансов.

Рассмотрим общую архитектуру протокола и взаимосвязь трех его составляющих. Одним из наиболее важных элементов схемы является требование аутентификации, которая производится на основе алгоритмов шифрования с открытым ключом. Это означает, что каждый участник взаимодействия (хост) должен иметь соответствующий ключ. Модель протокола допускает как вариант, когда один хост имеет множество ключей, так и вариант, когда множество хостов пользуются одним ключом. Для каждого из используемых алгоритмов хост должен иметь отдельный ключ (в качестве стандартного указан ключ DDS – Digital Signature Standard).

Ключ сервера используется для обеспечения уверенности клиента, что он работает с соответствующим сервером. Для этого клиент должен иметь некоторые предварительные знания о ключе сервера. Протокол допускает существование двух различных моделей обеспечения клиента такой информацией:

1. *Децентрализованная*. Клиент имеет базу данных открытых ключей хостов, ассоциированных с именем хоста. Преимущество модели в том, что она не требует инфраструктуры централизованного администрирования, недостаток – в том, что поддержание локальной базы данных требует дополнительных затрат.

2. *Централизованная*. Присутствует доверенная сторона – центр сертификации (certification authority – CA), который производит подтверждение ассоциации «хост/открытый ключ» для каждого ключа, полученного клиентом. Таким образом, клиенту достаточно хранить только ключ самого CA. С другой стороны, до начала взаимодействия каждый сервер должен быть сертифицирован CA. В настоящее время в Internet центры сертификации развертываются на основе стандартной инфраструктуры с открытым ключом PKI (Public Key Infrastructure).

Для конкретной реализации протокола необходимо учитывать ряд требований:

– алгоритмы шифрования, контроля целостности и сжатия должны быть отдельными для каждого из направлений, при этом должен быть определен предпочтительный алгоритм (первый из указанного в списке для каждой категории);

– для аутентификации хоста должны использоваться алгоритмы с открытым ключом и методы защищенного обмена ключами;

– сервер должен требовать аутентификации для каждого пользователя. Для некоторых или всех пользователей сервер может требовать множественную аутентификацию. При этом требуемые сервером алгоритмы аутентификации могут зависеть от месторасположения пользователя, пытающегося получить доступ;

– операции, которые разрешены пользователю к выполнению, определяются протоколом соединения. При этом должны быть четко определены допущения в работе, например такие, как запрет серверу инициировать сессии или исполнять команды на клиентской машине, а также установка соединения до его запроса клиентом. Кроме того, необходимо определить, кто и какие TCP-порты может запрашивать для соединения. Также необходимо учитывать, что соединение может проходить через межсетевой экран и пересекаться с его собственной политикой;

– алгоритмы шифрования, поддержания целостности и открытых ключей должны быть известны и проверены, длины ключей должны обеспечивать устойчивость к криптоанализу. Алгоритмы должны согласовываться, так чтобы в случае взлома одного можно было перейти на другой без модификации основного протокола.

4.4.1. Протокол транспортного уровня

В основу протокола транспортного уровня SSH положен протокол обмена ключами Диффи-Хеллмана, обеспечивающий одностороннюю аутентификацию в направлении от сервера к клиенту на основе цифровой подписи параметров ключа.

Напомним, что протокол обеспечивает аутентификацию только сервера, но не клиента, и использует зарегистрированный TCP-порт 22. Обмен ключами всегда инициируется клиентом, а сервер прослушивает конкретный порт, ожидая сеанса связи. При этом сервер может обслуживать несколько клиентов одновременно.

При описании протокола используются следующие обозначения:

C – клиент;

S – сервер;

p – большое безопасное простое число;

g – порождающий элемент подгруппы G_q поля $GF(p)$;

q – порядок подгруппы G_p;

V_C, V_S – версии протокола, принадлежащие клиенту и серверу соответственно;

K_S – открытый ключ сервера S;

I_C, I_S – первоначальное сообщение протокола обмена ключами, модифицированное перед началом протокола со стороны клиента и сервера соответственно.

Протокол обмена ключами выглядит следующим образом.

1. Клиент C генерирует случайное число x $(1< x< q)$, вычисляя значение

$$e \leftarrow g^x \pmod p,$$

и отсылает его серверу S.

2. Сервер S генерирует случайное число y $(0 < y < q)$ и вычисляет значение

$$f \leftarrow g^y \pmod p,$$

получает число e, вычисляя значения

$$K \leftarrow e^y \pmod p,$$

$$H \leftarrow hash\,(V_C \| V_S \| I_C \| I_S \| K_S \| e \| f \| K \|),$$

$$s \leftarrow \mathrm{Sig}_S(H)$$

и отсылает число $K_S \| f \| s$ клиенту C.

3. Клиент C убеждается, что число K_S действительно является ключом сервера S (используя любой подходящий метод, например, с помощью сертификата или проверенной локальной базы или метода). Затем клиент C находит числа

$$K \leftarrow f^x \pmod p,$$

$$H \leftarrow hash\,(V_C \| V_S \| I_C \| I_S \| K_S \| e \| f \| K \|),$$

и проверяет подпись s по числу H. Если аутентификация прошла успешно, клиент C принимает обмен ключами.

Протокол применяет следующие параметры (табл. 4.3).

Таблица 4.3

Параметры протокола SSH

Вид метода	**Наименование**	**Описание**	**Требование, поддержка**
Сжатие	none	Без сжатия	Обязательно
	zlib	GNU ZLIB (LZ77)	Опционально
Шифрование	3des-cbc	Трехключевой TripleDES в режиме CBC (168 бит)	Обязательно
	blowfish-cbc	Blowfish в режиме CBC (128 бит)	Рекомендовано
	twofish-cbc	Twofish в режиме CBC (256 бит)	Рекомендовано
	arcfour	Arcfour потоковый шифр	Опционально
	idea-cbc	IDEA в режиме CBC	Опционально
	cast128-cbc	CAST-128 в режиме CBC	Опционально
	none	Без шифрования	Не рекомендовано
Контроль целостности	hmac-sha1	HMAC-SHA1 длинной 160 битов	Обязательно
	hmac-sha-96	Первые 96 бит от HMAC-SHA1 HMAC-MD5 длиной 128 бит Первые 96 бит от HMAC-MD5	Рекомендовано
	hmac-md5	Без контроля	Опционально
	hmac-md5-96		Опционально
	none		Не рекомендовано
Обмен ключами	**diffie-hellman-group1-sha1**	**Метод Диффи-Хэллмана, комбинированный с электронной подписью**	**Обязательно**
Открытые ключи	ssh-dss	Обычный DSS (Digital Signature Standart)	Обязательно
	x509v3	Сертификаты X.509	Рекомендовано
	spki	Сертификаты SPKI	Опционально
	pgp	Сертификаты OpenPGP	Опционально

После обмена ключами все сообщения, которыми обмениваются обе стороны, будут зашифрованы с помощью согласованного сеансового ключа *K*. Затем обе стороны приступают к выполнению протокола аутентификации пользователя SSH.

4.4.2. Протокол аутентификации

Цель протокола – произвести аутентификацию клиента. Предполагается, что он будет работать поверх протокола транспортного уровня, который уже произвел аутентификацию сервера, установил шифрованное соединение и определил идентификатор сессии.

Для усложнения возможных атак перебора ключей аутентификации, сервер может временно блокировать свою работу после нескольких неудачных попыток аутентификации. Если при работе транспортного уровня была выбрана нерекомендованная опция работы без шифрования, тогда методы

аутентификации, основанные на передаче секретных данных, должны быть блокированы. Кроме того, при отсутствии контроля целостности необходимо отключить возможность изменения данных аутентификации, т.к. если атакующий изменит шифрованный поток данных это может привести к отказу в работе сервиса (атака класса deny-of-service).

Сервер управляет аутентификацией, указывая клиенту, какие виды аутентификации необходимы в какой период времени. Клиенту предоставляется возможность выбора из списка методов аутентификации сервера. Таким образом, с одной стороны, сервер контролирует методы аутентификации, с другой стороны – дает клиенту возможность выбрать поддерживаемый им метод.

По протоколу клиент выставляет запрос, цель которого – получение списка методов аутентификации от сервера. Если после отправления списка методов аутентификации сервером клиент не отвечает, сервер имеет возможность разорвать соединение по тайм-ауту (рекомендуемый период – 10 мин).

4.4.3. Протокол соединения

Предполагается, что протокол функционирует поверх транспорта, обеспечивающего безопасность путем шифрования, контроля целостности и аутентификации. Протокол используется для выполнения команд на удаленной машине, а также может быть использован сервером клиентской станции для запуска команд исполнение (это может быть запрещено в отдельных реализациях для снижения уровня риска атаки). Протокол подробно описывает форматы различных сообщений о выполнении команд, таких, как открытие и закрытие канала, передача данных, продвижение и т.п.

4.5. Протокол SOCKS, версия 5

Протокол SOCKS [RFC-1928] предназначен для прозрачного и безопасного прохождения сложных протоколов прикладного уровня через межсетевой экран типа «посредник приложений». Одним из основных требований к безопасности в этом смысле является строгая аутентификация. SOCKS5 завоевал широкую популярность в Интернете из-за простоты своего функционирования в сочетании с широкой универсальностью выполняемых функций. Протокол разработан для обеспечения функционирования клиент/серверных приложений поверх TPC или UDP и работает как прослойка между прикладным и транспортным уровнями, не предоставляя никаких сервисов сетевого уровня. Протокол различает работу TPC- и UDP-клиентов.

Когда TPC-клиент желает установить соединение с сервером, доступным только через межсетевой экран, он должен установить соединение с управляющим портом SOCKS-сервера (номер порта, зарегистрированный IANA, – 1080). В рамках управляющего соединения сервер обрабатывает запрос (рис. 4.7) и производит следующие действия:

– согласовывает метод аутентификации клиента;

– аутентифицирует клиента выбранным методом;

– в случае необходимости производит трансляцию имени удаленного сервера в сетевой адрес (например, DNS-имени в IP-адрес).

VER	NMETHODS	METHODS
1 октет	1 октет	от 1 до 255 октетов

Рис. 4.7. Формат аутентификационного запроса клиента SOCKS-серверу

Затем SOCKS-сервер устанавливает два соединения вне управляющего соединения для передачи данных:

– первое соединение – между собой и удаленным сервером. В рамках данного соединения сервер от своего имени будет пересылать все запросы клиента и принимать ответы на них; порт SOCKS-сервера в сторону удаленного ресурса выбирается произвольно динамически, и в случае необходимости сообщается клиенту.

– второе соединение – между собой и клиентом. В рамках этого соединения клиентом будут передаваться запросы к удаленному серверу и возвращаться SOCKS-сервером ответы на них удаленного ресурса. Порт SOCKS-сервера в сторону клиента выбирается произвольно динамически и всегда сообщается клиенту (в рамках управляющего соединения).

Как и в любой прокси-технологии, клиент никогда не производит соединения непосредственно с удаленным сервером. Все данные, как в прямом, так и в обратном направлении проходят через SOCKS-сервер. Кроме того, поскольку аутентичность клиента является одной из наиболее важных функциональных нагрузок протокола SOCKS5, в нем заложена жесткая привязка к управляющему соединению. Как только случайно или преднамеренно разрывается соединение клиента с управляющим портом, SOCKS-сервер сразу же перестает транслировать данные как в прямом, так и в обратном направлении во всех установленных соединениях данного клиента. Это касается как TPC, так и UDP-соединений.

Поле VER указывает номер версии протокола (для текущей версии = 05).

Поле NMETHODS указывает количество октетов в поле METHODS.

Поле METHODS принимает следующие значения:

00 – не требует аутентификации;

01 – аутентификация GSS-API [RFC-1961];

03 – аутентификация «имя пользователя/пароль» [RFC-1929];

02-7F – методы аутентификации IANA;

08-FE – собственные реализации методов аутентификации;

FE – нет приемлемых методов аутентификации – возвращается сервером, если клиент не может предоставить для него допустимые методы аутентификации.

Для обеспечения совместимости реализация протокола обязательно должна поддерживать аутентификацию GSS-API [RFC-1961] и желательно – аутентификацию «имя пользователя/пароль» [RFC-1929].

После аутентификации клиент направляет серверу запрос на соединение (рис. 4.8).

VER	CMD	RSV	ATYP	DST.ADDR	DST.PORT
1 октет	1 октет	00	1 октет	Зависит от ATYP	2 октета

Рис. 4.8. Формат запроса на соединение клиента SOCKS-серверу

Поле VER указывает номер версии протокола (для текущей версии = 05).

Поле CMD – вид запроса:

Connect – соединение (CMD = 1);

Bind – связь (CMD = 2);

UDP-Associate – UDP-ассоциация (CMD = 3).

Поле RSV зарезервировано (= 00).

Поле ATYP – тип адреса в поле DST.ADDR – адрес назначения.

Поле DST.PORT – порт назначения.

SOCKS-сервер после обработки клиентского запроса возвращает клиенту следующий ответ (рис. 4.9).

VER	REP	RSV	ATYP	BND.ADDR	BND.PORT
1 октет	1 октет	00	1 октет	Зависит от ATYP	2 октета

Рис. 4.9. Ответ SOCKS-сервера

Поле VER указывает номер версии протокола (для текущей версии = 05).

Поле REP – код ответа:

00 – успешно;

01 – неклассифицированная ошибка SOCKS-сервера;

02 – соединение не разрешено правилами;

03 – сеть недоступна;

04 – хост недоступен;

05 – в соединении отказано;

06 – TTL (time-to-line) время до удаленного сервера истекло;

07 – команда не поддерживается;

08 – тип адреса не поддерживается;

09 – FE – не назначены.

Поле RSV зарезервировано (= 00).

Поле ATYP указывает тип адреса в поле BND.ADDR, адрес составлен из следующих значений:

IP-адрес версии 4 (ATYP = 01);
доменное имя (ATYP = 03);
IP-адрес версии 6 (ATYP = 04).

Поля BND.ADDR и DST.PORT представляют собой адрес и порт SOCKS-сервера, на котором он открыл какое-либо соединение (к клиенту или серверу). Они имеют разную смысловую нагрузку в зависимости от типа запрашиваемого клиентом соединения.

Если в запросе указан тип Connect, то происходит «активное подключение» (клиент инициирует сессию к удаленному ресурсу). В этом случае поля BND.ADDR и DST.PORT содержат адрес и порт, на которых SOCKS-сервер открыл соединение в сторону клиента. В рамках этого соединения SOCKS-сервер будет транслировать все данные, пришедшие от удаленного сервера. При запросе типа Connect адрес и порт соединения, открытого SOCKS-сервером в сторону удаленного ресурса, остаются для клиента неизвестными.

Если в запросе указан тип BIND, то это означает, что подключение должно было произойти по пассивной схеме. Если бы не было сетевого экрана, клиент должен был бы открыть на ожидание сессии произвольный порт, сообщить его номер удаленному серверу, а уже тот – инициировать сеанс на указанный номер порта клиента. Классическим примером протокола, требующего по умолчанию для передачи данных именно такой тип соединения, является FTP. Однако межсетевой экран, если он установлен между клиентом и сервером, не позволит удаленному хосту инициировать соединения на произвольный порт клиента.

С помощью SOCKS-сервера необходимое соединение происходит следующим образом.

1. Клиент оправляет SOCKS-серверу запрос типа Bind.
2. SOCKS-сервер открывает в сторону удаленного ресурса динамически произвольный порт в режиме ожидания соединения и сообщает клиенту в рамках управляющей сессии в первом своем ответе адрес и порт открытого соединения (поля BND.ADDR и DST.PORT).
3. Клиент формирует запрос прикладного уровня, в котором требовалось указать порт пассивного соединения, но подставляет туда не свой адрес, а присланные от SOCKS-сервера параметры.
4. Как только в адрес порта, открытого SOCKS-сервером в режиме ожидания, приходит первый пакет от удаленного ресурса (запрос на инициирование соединения), SOCKS-сервер высылает в рамках управляющей сессии второй ответ клиенту, указывая в полях BND.ADDR и DST.PORT адрес и порт, открытые им для этого потока уже в сторону клиента.
5. После установления всех соединений SOCKS-сервер начинает прозрачно транслировать данные в обоих направлениях подобно тому, как если бы соединение устанавливалось по типу Connect.

Для запроса UDP-Associate, используемого для пересылки UDP-дейтаграмм через межсетевой экран, в полях BND.ADDR и DST.PORT ответа SOCKS-сервер указывает свои адрес и порт, куда клиент обязан посылать UDP-запросы, которые должны пересылаться удаленному ресурсу.

Клиент должен направлять свои дейтаграммы SOCKS-серверу, производя инкапсуляцию (сокрытие значимых данных) дейтаграммы, если этого требует выбранный метод аутентификации. Каждая UDP-дейтаграмма должна быть снабжена следующим дополнительным заголовком (рис. 4.10).

RSV	FRAG	ATYP	DST.ADDR	DSP.PORT	DATA
2 октета	1 октет	1 октет	Зависит от ATYP	2 октета	Данные

Рис. 4.10. UDP-запросы

Поле RSV зарезервировано (= 0000).

Поле FRAG – номер текущего фрагмента, если данные фрагментированы.

Поле ATYP – тип адреса в поле BND.ADDR состоит из следующих значений:

IP-адрес версии 4 (ATYP = 01);

доменное имя (ATYP = 03);

IP-адрес версии 6 (ATYP = 04);

Поле BND.ADDR – требуемый адрес назначения.

Поле DST.PORT – требуемый порт назначения.

Поле DATA – данный пользователь.

При открытии клиентом вторичных соединений включение в ответ SOCKS-сервера поля BND.ADDR, а не первоначального адреса управляющей сессии, позволяет создавать целый кластер прокси-серверов, работающих под управлением мастер-сервера. В этом случае мастер-сервер поддерживает только управляющие сессии со всеми подключенными к кластеру клиентами, а вся нагрузка на собственно трансляцию потоков данных (т.е. вторичные соединения) распределяется между несколькими «рабочими» серверами.

При обработке каждого очередного запроса от клиента мастер-сервер по какому-либо правилу выбирает один «рабочий» сервер и обменивается с ним всей необходимой информацией о запросе. Затем в ответе клиенту в поля BND.ADDR и DST.PORT подставляются уже координаты «рабочего» сервера трансляции.

4.6. Инфраструктура открытых ключей

Для того чтобы два субъекта могли безопасно обмениваться информацией по незащищенным каналам связи, используются различные методы и алгоритмы криптографии, в том числе и алгоритмы с открытыми ключами.

Общий смысл подобных алгоритмов состоит в том, что каждый субъект создает пару связанных ключей – секретный и открытый.

Предполагая в этой ситуации, что секретный ключ сохраняется субъектом в строгом секрете, а открытый ключ распространен между остальными участниками информационного обмена, можно решать различные вопросы безопасности связанные с аутентификацией. Например, при помощи электронно-цифровой подписи (ЭЦП) под сообщением субъекта на основе секретного ключа можно обеспечить аутентификацию отправителя (только тот, кто имеет секретный ключ, может установить истинную ЭЦП под присланным текстом) и необходимые условия для защиты от отказа передачи данных (только тот, кто имеет секретный ключ, мог создать текущий набор данных).

При этом вся схема безопасности работает только при одном важном условии – если получатель информации обладает действительно тем самым открытым ключом, секретный ключ для которого находится только у отправителя. Это означает:

– что секретный ключ, соответствующий открытому ключу, не должен быть скомпроментирован;

– что открытый ключ, соответствующий секретному ключу, доставлен участнику процесса информационного обмена надежным способом, т.е. не произошла его подмена ключом из другой пары, принадлежащей злоумышленнику.

Именно для решения второй задачи и развертывается инфраструктура открытых ключей PKI. В системе PKI вводятся два важных понятия – цифровой сертификат и центр сертификации (CA). CA – это организация, которой априори доверяют все участники информационного взаимодействия и которая выпускает цифровые сертификаты, обеспечивающие уверенность в том, что данный открытый ключ действительно принадлежит заявленному субъекту.

Цифровой сертификат – это электронный документ, состоящий из следующих частей:

– идентификатора владельца сертификата (уникально определяющего его как в реальном, так и в виртуальном пространстве);

– открытого ключа владельца сертификата;

– данных самого сертификата, например, срока действия;

– ЭЦП CA, подтверждающей, что указанные данные проверены и соответствуют действительности.

Каким образом открытый ключ субъекта будет надежно доставлен до CA и как CA проверяет корректность данных субъекта, – это вопрос политики формирования сертификатов CA. Однако после того, как получен корректный цифровой сертификат, его распространение между участниками будет очень простым. Субъект может публиковать свой сертификат на своем сервере, автоматически высылать его каждому участнику информационного обмена либо при инициализации каждой новой сессии поручить CA предоставлять свой сертификат по запросу пользователей.

При такой централизованной модели хорошо внедряется иерархическая система доверия. Если субъект имеет свой цифровой сертификат, подписанный CA, которому широко доверяют, то он сам может выступить в качестве CA для ряда подчиненных или независимых субъектов. В свою очередь каждый из этих субъектов, имея свой сертификат, подтвержденный CA *n*-го уровня, о котором сам участник ничего не знает, должен подниматься вверх по иерархии, получая все сертификаты до того CA, которому доверяет данный участник (вплоть до первого CA).

Для любой схемы распространения открытых ключей необходимо выполнение следующих функций:

– регистрация, т.е. первичное информирование CA о существовании субъекта, чей открытый ключ подлежит сертификации;

– инициализация, получение клиентами CA цифрового сертификата самого CA и другие процедуры;

– сертификация, т.е. формирование цифрового сертификата субъектов с его открытым ключом;

– генерация пары ключей, которая может быть сделана самим субъектом, либо по договоренности с CA, который может сгенерировать ключи для субъекта;

– восстановление ключей в случае их утраты субъектом, CA может восстановить ключи из своего доверенного хранилища;

– обновление ключей, чтобы свести к минимуму риск реализации атак по взлому ключей (соответственно цифровые сертификаты также должны быть обновлены);

– перекрестная сертификация CA, принадлежащих к различным доменам ответственности, может понадобиться, когда их пользователям необходимо работать с ними одновременно;

– отзыв сертификатов, реализуемый использованием механизма Списка отозванных сертификатов (Certificate revocation List – CRT), который CA распространяет среди пользователей.

Иллюстрация жизненного цикла ключей приведена на рис. 4.11.

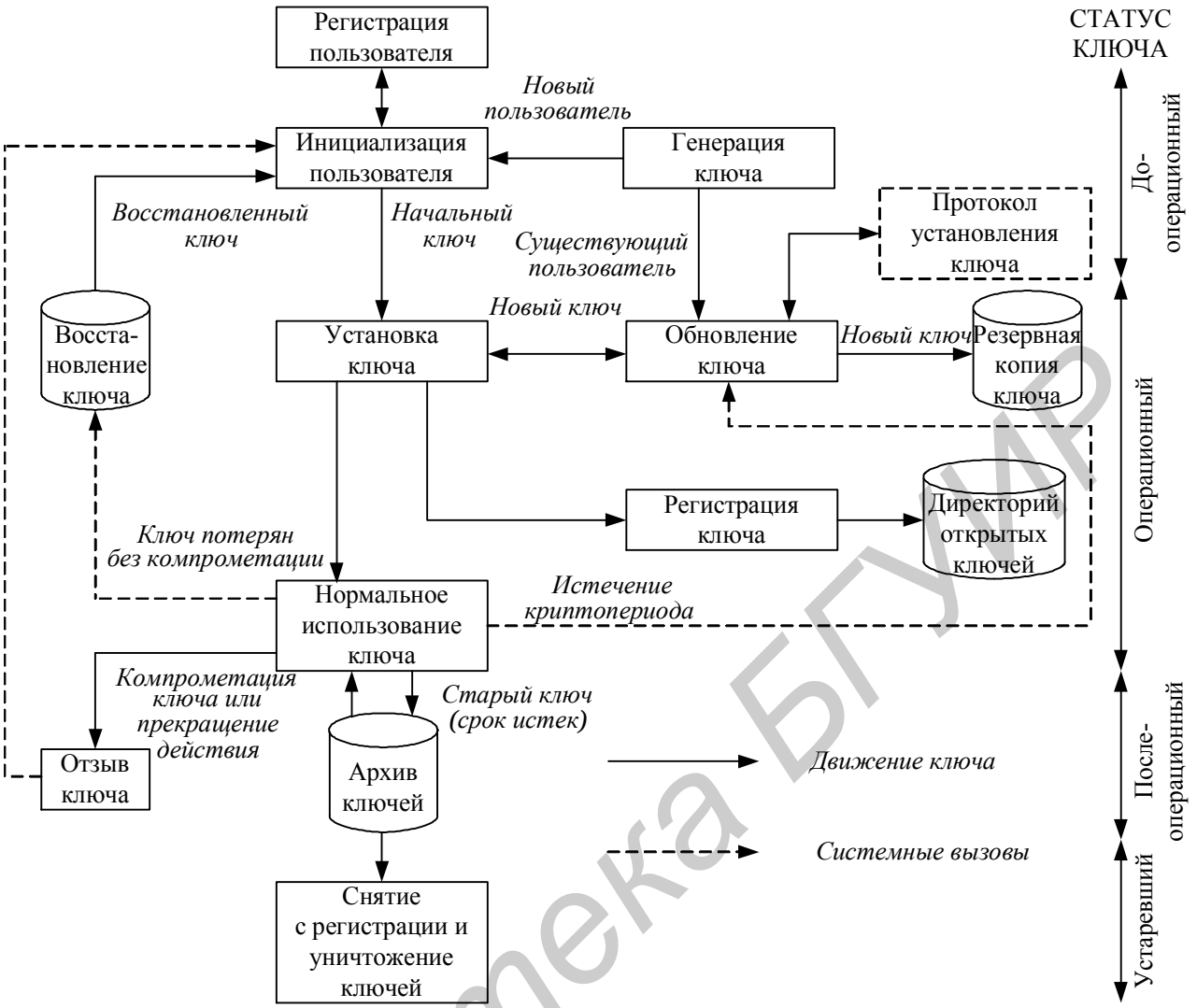

Рис. 4.11. Жизненный цикл ключей

4.7. Цифровые сертификаты X.509 v3

Рассмотрим подробнее структуру и формат самого сертификата X.509 как одного из наиболее широкого используемых протоколов (рис. 4.12). Различным аспектам данного вопроса посвящено большое количество документов (RFC 2459, 2510, 2511, 2527, 2528, 2559, 2560, 2585, 2587, 3029, 3039).

Поле **Номер версии** определяет номер версии используемого сертификата. Равен 0 (версия 1), если присутствуют только базовые поля, без расширений и идентификаторов субъекта/изготовителя. Если присутствуют идентификаторы – равен 1 (версия 2), а если присутствует расширение – равен 2 (версия 3).

Поле **Серийный номер** – уникальный номер, присваиваемый каждому сертификату.

Поле **Алгоритм подписи** – идентификатор алгоритма, используемый при подписании сертификата. Должен совпадать с полем **Алгоритм ЭЦП**.

Поле **Изготовитель** – идентифицирует того, кто выпустил и подписал сертификат, т.е. фактически сертифицирующего уполномоченного CA. Формат данного поля представляет относительное отдельное имя (Relative Distinguished Name), используемое, например, в службах и протоколах каталогов (X.500 или LDAP) и состоящее из комбинации различных параметров, таких, как идентификатор страны, области, организации, домен, субдомен, фамилия/имя (или наименование), дающих в совокупности некий уникальный идентификатор. Стандартный набор таких параметров или атрибутов определен в протоколе X.500.

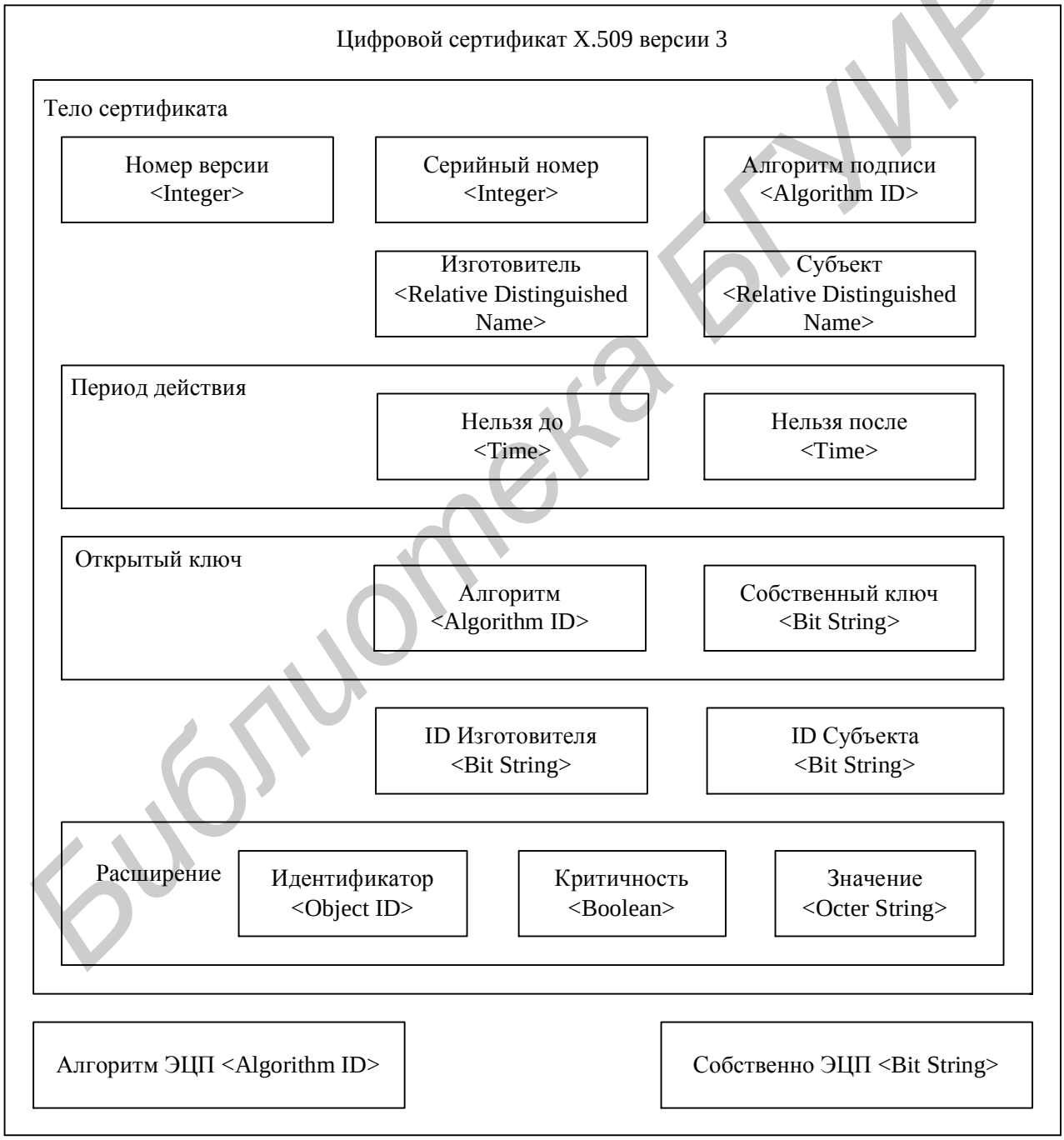

Рис. 4.12. Формат сертификата X.509 версии 3

Поле **Субъект** – идентифицирует владельца сертификата: того, кому принадлежит открытый ключ, указанный в сертификате. Формат также представлен в виде относительного отдельного имени.

Поле **Период действия** – временной интервал, в течении которого СА гарантирует поддержку статуса сертификата (например своевременное сообщение о его отзыве). Представлено двумя полями – временем начала и завершением периода.

Поле **Открытый ключ** – два поля, первое из которых указывает, какой алгоритм был использован для генерации ключа, а второе представляет собственно открытый ключ в виде набора битов.

Поле **Идентификаторы субъекта и изготовителя** указаны на случай возникновения повторного использования одинакового имени субъекта или изготовителя.

Поле **Расширение** – это дополнительный атрибут, связанный с субъектом, изготовителем или открытым ключом и предназначенный для управления процессами сертификации.

Поле **Алгоритм ЭЦП** – идентификатор алгоритма, используемый при подписании сертификата. Должен совпадать с полем **Алгоритм подписи**.

Собственно поле ЭЦП – набор битов, составляющих электронно-цифровую подпись под данным сертификатом. В ее формировании участвуют поля, указанные в теле сертификата.

Отдельное внимание в описании протокола уделено возможным расширениям. В список основных функциональных расширений входят:

– идентификация конкретной пары «открытый/секретный ключ» изготовителя сертификатов, в случае если изготовитель использует несколько различных ключей для подписания различных сертификатов;

– идентификация конкретного открытого ключа субъекта, в случае если субъект имеет несколько сертификатов, возможно, от разных изготовителей;

– цель использования ключа – для шифрования, подписи, формирования других сертификатов;

– уточнение периода использования – можно сократить период действия сертификатов, указанных в поле **Период действия**;

– уточнение политики использования сертификата – указывается, для каких целей выпущен сертификат и приложения, использующие сертификат; могут принимать или отвергать этот сертификат на основе этого расширения;

– выбор соответствия политик использования сертификатов для изготовителя и для субъекта, если имеются различные варианты;

– альтернативное имя субъекта или изготовителя, если оно имеется и отличается от указанного в основной части сертификата;

– определение того, является ли сам субъект СА и насколько глубоко разворачивается цепочка сертификатов через данный СА.

Рассмотрим теперь структуру списка отозванных сертификатов (рис. 4.13).

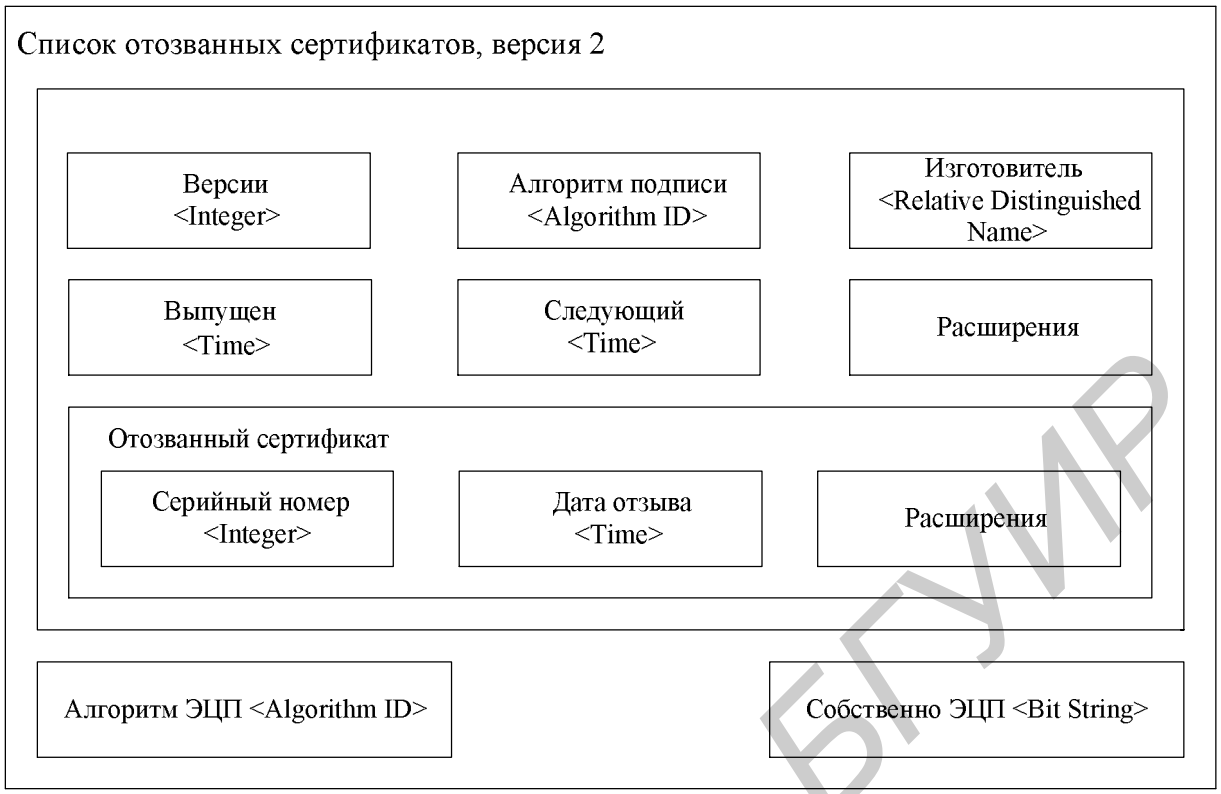

Рис. 4.13. Структура Списка отозванных сертификатов

Поле **Номер версии** определяет номер версии используемого CRL. Текущая используемая версия – вторая, она обязательна для указывания, если CRL используется с расширениями.

Поле **Алгоритм подписи** – идентификатор алгоритма, используемый при подписании сертификата. Должен совпадать с полем Алгоритм ЭЦП.

Поле **Изготовитель** – идентифицирует того, кто выпустил и подписал CRL. Формат данного поля представляет относительное отдельное имя, используемое, например, в службах и протоколах каталогов (X.500 или LDAP) и состоящее из комбинации различных параметров, таких как идентификатор страны, области, организации, домен, субдомен, фамилия/имя (или наименование), дающих в совокупности некий уникальный идентификатор. Стандартный набор таких параметров или атрибутов определен в протоколе X.500.

Поле **Выпущен** – указывает время выпуска настоящего CRL.

Поле **Следующий** – указывает время выпуска следующего CRL.

Поле **Расширение** – это дополнительный атрибут, связанный с пользователем или открытым ключом и предназначенный для управления процессами сертификации.

Поле **Отозванный сертификат** – таких полей может быть несколько – указывает, сколько сертификатов отзывается. Об отозванном сертификате приводится следующая информация:

– серийный номер;

– дата вступления отзыва в силу.

– расширения отозванного сертификата. Они включают:

Поле **Алгоритм ЭЦП** – идентификатор алгоритма, используемый при подписании сертификата. Должен совпадать с полем **Алгоритм подписи**.

Собственное поле ЭЦП – набор битов, составляющих электронно-цифровую подпись под данным сертификатом. В ее формировании участвуют поля, указанные в теле CRL.

Расширения самого CRL включают схожую с сертификатом функциональность, а также дополнительную функциональность:

– обеспечивают последовательную нумерацию CRL для облегчения управления списками;

– обеспечивают накопление приложениями информации об отозванных сертификатах (так называемые delta CRL);

– определяют пункт распространения CRL.

Расширения, связанные с отозванным сертификатом, раскрывают причину отзыва сертификата, определяют фиксированный набор инструкций – действий, выполняемых с отзываемым сертификатом, – указывают дату компроментации секретного ключа или другой причины недействительности сертификата и т.д. Полный список расширений списка отозванных сертификатов можно найти в соответствующей документации.

Контрольные вопросы и задачи

1. Предположим, что соединения в Internet не защищены протоколом ISec. Каким образом нарушитель может манипулировать сообщениями, передаваемыми через Internet (например, имитировать автора, перенаправлять сообщение и т.п.)?

2. Какую роль играет заголовок аутентификации (AH) в протоколе PSec?

3. Назовите два способа криптографической защиты IP-пакета.

4. В протоколе IKE используются асимметричные криптографические методы. Можно ли утверждать, что этот протокол является протоколом аутентификации с открытым ключом?

5. Что такое идентификатор «актуальности»?

6. Предположим, что криптографическая операция была выполнена недавно. Гарантирует ли это «актуальность» сообщения, посланного пользователем?

7. Почему в протоколе TLS открытый ключ, созданный по закрытому ключу с помощью необратимой функции, должен быть сертифицирован?

8. Приводит ли аннулирование сертификата открытого ключа к отзыву цифровой подписи, использованной до момента аннулирования?

9. Какой протокол обмена ключами положен в основу протокола транспортного уровня SSH?

10. Что такое инфраструктура открытых ключей? Поясните назначение цифровых сертификатов X.509.

ГЛАВА 5. ОЦЕНКА ПРОТОКОЛОВ АУТЕНТИФИКАЦИИ

Как правило, крупные сети, состоящие из компьютеров и других устройств, являются открытыми. Это значит, что пользователи, в роли которых могут выступать человек, компьютер, устройство, ресурс, провайдер (или все они в совокупности), могут присоединяться к сети, чтобы передавать и получать сообщения от других пользователей, входящих в сеть, без разрешения «главного» администратора доступа. Установление подлинности участников взаимодействия является задачей протоколов аутентификации, которая решается на основе применения криптографических методов. В открытой среде всегда имеют место атаки на протокол аутентификации, которые организуются противником (коалицией противников), преследующих незаконные цели. Цель противника может быть серьезной, например, взлом секретного сообщения или ключа, или менее серьезной, например, обман получателя сообщения. Как правило, протокол аутентификации считается некорректным, если пользователь считает, что протокол выполняется правильно и связь установлена с подлинным партнером, в то время как подлинный партнер приходит к противоположному выводу.

Следует подчеркнуть, что атаки на протоколы аутентификации, как правило, не связаны со взломом криптографических алгоритмов. Обычные протоколы аутентификации небезопасны не потому, что в них применяются слабые криптографические алгоритмы, а потому, что имеют недостатки, позволяющие противнику пройти аутентификацию, не прибегая ко взлому криптографического алгоритма. По этой причине при анализе протоколов аутентификации обычно предполагают, что лежащий в основе протокола криптографический алгоритм является стойким, и не рассматривают его потенциальные слабости (анализ слабых мест криптографических алгоритмов является предметом других дисциплин).

5.1. Формальные методы анализа протоколов аутентификации

Предполагая существование мощных средств воздействия в такой уязвимой среде, как открытая сеть, при оценке протоколов аутентификации широко используется следующая модель нарушителя. В рамках этой модели нарушитель обладает следующими характеристиками:

– может перехватить любое сообщение, передаваемое по сети;

– является законным пользователем сети и может вступать в контакт с любым другим пользователем;

– может получать сообщения от любого пользователя;

– может посылать сообщения любому пользователю, маскируясь под любого другого пользователя.

Таким образом, в данной модели любое сообщение, передаваемое по сети, оказывается в распоряжении нарушителя. Следовательно, существует

угроза, что любое сообщение, полученное пользователем из сети, прежде было перехвачено нарушителем, а затем передано по адресу. Иначе говоря, предполагается, что нарушитель имеет полный контроль над всей сетью.

Однако несмотря на перечисленные выше возможности, считается, что существует ряд ограничений, которые нарушитель не в состоянии преодолеть. К ним относятся следующие:

– нарушитель не может угадать случайное число, выбранное из достаточно большого пространства;

– не имея правильного секретного ключа, нарушитель не может восстановить открытый текст по его зашифрованному варианту и наоборот, не может правильно зашифровать исходное сообщение с помощью идеального алгоритма шифрования;

– нарушитель не способен найти секретный компонент, т.е. секретный ключ, соответствующий заданному открытому ключу;

– контролируя средства связи и большую открытую часть вычислений пользователя, нарушитель не имеет доступа ко многим закрытым зонам вычислительной среды, например, к памяти вычислительного устройства автономного пользователя.

При анализе стойкости протоколов принимают также следующие соглашения о поведении законных и незаконных участников протокола.

1. Подлинный пользователь, участвующий в протоколе, не понимает семантического смысла ни одного протокольного сообщения, пока протокол не завершится успешно.

2. Подлинный пользователь, участвующий в протоколе, не способен ни распознать, ни создать, ни разложить сообщение на составные части, не имея правильного ключа.

3. Подлинный пользователь, участвующий в протоколе, не способен распознавать псевдослучайные числа, порядковые номера и криптографические ключи, если они не сгенерированы самим пользователем в рамках текущего сеанса или не являются результатом выполнения протокола.

4. Подлинный пользователь, участвующий в протоколе, не записывает протокольные сообщения, если этого не требует спецификация протокола. Как правило, протокол аутентификации не имеет истории, т.е. пользователь не должен запоминать никакой информации о состоянии, возникшем после успешного выполнения протокола, за исключением информации, являющейся результатом выполнения протокола и предназначенной для самого пользователя.

5. Нарушитель (в дополнение к описанной выше модели) знает о наивности (готовности подыграть) подлинных пользователей и всегда пытается использовать это обстоятельство.

Протокол аутентификации – это не только алгоритм, но и процедура обмена данными между разными участниками в виде сообщений по компьютерным сетям в соответствии с установленными правилами. Таким образом, существует еще один аспект эффективности протоколов

аутентификации: количество сеансов связи, которые часто называют раундами. Обычно сеанс связи считается более дорогим, чем этап локальных вычислений (как правило, этот этап состоит из выполнения компьютерных инструкций, например, умножения двух чисел). Поэтому желательно минимизировать количество раундов в протоколе. В соответствии со стандартным критерием качества алгоритм считается эффективным, если время его выполнения ограничено полиномом невысокой степени, зависящим от размера задачи. Аналогично протокол считается эффективным, если количество раундов связи ограничено полиномом малой степени: константой (степень равна 0) или линейной функцией (степень равна 1). Протокол, в котором количество раундов превышает линейную функцию, нельзя признать практически эффективным.

В области формального анализа протоколов аутентификации существует ряд методов (их классификация приведена на рис. 5.1), при этом различаются два основных подхода. В первом используются формальные выводы о некоторых полезных свойствах, например, стойкости, а во втором осуществляется поиск нежелательных и опасных дефектов.

В рамках первого подхода анализируемый протокол должен выбираться (или разрабатываться) весьма тщательно. Анализ должен показать, что выбранный протокол действительно соответствует предъявляемым требованиям, которые также должны быть строго формализованы.

Рис. 5.1. Классификация методов анализа протоколов аутентификации

Этот подход основывается на использовании вычислительных методов и методов логики. При использовании вычислительных методов свойства стойкости стремятся выразить с помощью измерения вероятности, а исследование правильности протокола сводится к доказательству определенной

теоремы. Довольно часто в доказательствах стойкости используется метод редукции атаки для решения общепринятых трудноразрешимых задач из теории вычислительной сложности. При использовании методов логики свойства стойкости выражают в виде множества абстрактных символов, допускающих определенные манипуляции, иногда путем применения логических правил, иногда – с помощью шаблонных средств для доказательства теорем. Результатом этих манипуляций должен быть ответ ДА или НЕТ.

В рамках второго подхода считается, что любой протокол аутентификации, выбранный наугад, тщательно разработанный или имеющий математическое доказательство стойкости, все равно может содержать ошибку. Ведь формально доказательство правильности демонстрирует лишь соответствие протокола множеству установленных желательных свойств. Вполне возможно, что такой протокол окажется скомпрометированным, если его дефект в процессе доказательства правильности не был учтен. Таким образом, данный подход направлен на исчерпывающий анализ всех возможных ошибок. Для формализации протокол представляется в виде системы с конечным множеством состояний, которую часто разбивают на частичные протоколы, выполняемые разными пользователями (включая нарушителя). Этот подход тесно связан с формальным анализом сложных систем, поэтому для исследования протоколов аутентификации часто применяются автоматизированные средства анализа.

5.2. Вычислительные модели доказательства корректности протоколов

Формальное доказательство стойкости в рамках вычислительной модели состоит из трех этапов.

1. Описание формального поведения участников протокола и атакующего алгоритма: моделирование, как правило, осуществляется с помощью атакующей игры, в которой участвуют атакующий алгоритм и объект атаки.

2. Формальное определение стойкости: выигрыш атакующего алгоритма в атакующей игре обычно выражается через вероятность (значимую) и оценку временной сложности (разумной).

3. Формальная демонстрация существования полиномиальной редукции, сводящей атаку к решению трудноразрешимой задачи, которая, как правило, имеет вид математической теоремы.

Рассмотрим данный метод на примере доказательства стойкости протокола двусторонней аутентификации пользователей с использованием общего симметричного ключа.

5.2.1. Формальное моделирование поведения участников протокола

Описание протокола состоит из двух частей. Во-первых, каждый из двух участников протокола моделируется эффективным исполняемым кодом, имеющим входную и выходную информацию. Во-вторых, сеансы связи между двумя участниками представляются в виде совокупности связей.

Следует подчеркнуть, что в рассматриваемых сеансах связи участвует нарушитель, который может манипулировать значениями, передаваемыми по каналу связи.

5.2.2. Формализация части протокола, выполняемой подлинным участником

С формальной точки зрения абстрактный протокол описывается функцией V, имеющей полиномиальную сложность и следующие аргументы:

– k – параметр безопасности $k \in N$;

– i – идентификатор пользователя $i \in I$, выполняющего данную часть протокола. Будем называть этого пользователя «владельцем». Множество I состоит из пользователей, обладающих одним и тем же долговременным ключом;

– j – идентификатор партнера, с которым общается владелец; $j \in I$;

– K – долговременный симметричный ключ (т.е. секретная входная информация). В двустороннем протоколе симметричный ключ K принадлежит как владельцу, так и его партнеру;

– $conv$ – предыдущие сообщения, представляющие собой строку битов ($conv$ от англ.: conversation). Эта строка увеличивается по мере выполнения протокола, причем новая строка приписывается к старой;

– r – случайный аргумент владельца, являющийся одноразовым случайным числом, генерируемым владельцем.

Поскольку $P(k, i, j, K, conv, r)$ имеет полиномиальную сложность, зависящую от размера аргументов (отметим, что размер параметра k равен k бит), можно считать, что аргументы K и r имеют размер k, а размер аргументов r, j и $conv$ полиномиально зависит от параметра k.

Значениями функции $P(k, i, j, K, conv, r)$ являются три числа:

– m – следующее сообщение, подлежащее отправке, – $m \in \{0,1\}^* \cup \{$«сообщений нет»$\}$. Это открытое сообщение, подлежащее отправке адресату через открытую сеть;

– $_д$ – решение пользователя – $d \in \{$Принять, Отказать, Не принимать решения$\}$. Пользователь решает, принять или отвергнуть идентификатор партнера по переговорам, либо не принимать решения вообще. Принятие идентификатора, как правило, откладывается до завершения протокола, а отклонить его можно в любой момент. Если пользователь принимает какое-либо определенное решение, значение $_д$ больше не изменяется;

– – закрытый результат, вычисленный владельцем, – $\in \{0,1\}^* \cup \{$«результата нет»$\}$. В данном случае закрытым результатом, вычисленным

владельцем при благоприятном исходе протокола, является согласованный сеансовый ключ.

Анализ показывает, что в формальную модель входят основные компоненты протоколов аутентификации: криптографические операции, идентификаторы участников, идентификаторы «актуальности» и сами сообщения.

5.2.3. Формализация обмена информацией

Для любых двух участников протокола, обладающих одним и тем же долговременным симметричным ключом и для числа $s \in N$ обозначим через $\prod_{i,j}^{s}$ абонента *i*, пытающегося аутентифицировать абонента *j* в ходе сеанса с меткой *s*. Эта попытка может инициироваться абонентом *i*, а может быть и ответом на сообщение, полученное от абонента *j*. При анализе протокола такая попытка всегда интерпретируется как ответ подыгрывающего на запрос, посланный нарушителем.

Нарушителю, контролирующему сеть, могут быть известны последовательности абонентов $\prod_{i,j}^{s}$ и $\prod_{j,i}^{t}$ для любой пары *i*, *j*, обладающей общим долговременным симметричным ключом, даже если он не вычисляет их сам. Однако будучи активным атакующим алгоритмом, нарушитель может организовывать сколько угодно сеансов связи и убеждать любого пользователя (например *i*) начать протокол, считая его другим подлинным пользователем (например *j*).

Поскольку нарушитель является мощным и активным атакующим алгоритмом, он может использовать игроков $\prod_{i,j}^{s}$ и $\prod_{j,i}^{t}$ ($i, j \in I, s, t \in N$) в качестве подыгрывающих, представляющих собой «черные ящики». Это значит, что нарушитель может послать запрос подыгрывающему $\prod_{i,j}^{s}$ передавая пользователю *i* аргументы (*i*, *j*, *s*, *conv*). Аналогичным образом нарушитель может послать запрос подыгрывающему $\prod_{i,j}^{t}$. Когда нарушитель посылает запрос подыгрывающему $\prod_{i,j}^{s}$ используя аргументы (*i*, *j*, *s*, *conv*), пользователь *i* добавляет к ним собственный секретный ключ *K*, случайное число *r* и вычисляет функцию $\prod^{s}$ (1^k, *i*, *j*, *K*, *conv*, *r*). После этого пользователь *i* отсылает в сеть результат *m* (если он существует) или строку «сообщений нет», а также решение d, сохраняя у себя закрытый компонент результата *a*. Все результаты, посланные пользователем в сеть, оказываются в распоряжении нарушителя и могут быть использованы для организации атаки.

Можно считать, что среди нарушителей всегда есть относительно безопасный противник, называемый «безвредным». Он просто выбирает пары

подыгрывающих $\prod_{i,j}^{s}$ и $\prod_{j,i}^{t}$, а затем перехватывает и точно передает сообщения, которыми они обмениваются, начиная с подыгрывающего $\prod_{i,j}^{s}$. Другими словами, первый запрос безвредного нарушителя выглядит так: (*i*, *j*, *s*, "") (здесь символы "" обозначают пустую строку). В ответ он получает сообщение $m_1^{(i)}$. На второй запрос (*j*, *i*, *t*, $m_1^{(i)}$) нарушитель получает сообщение $m_1^{(j)}$ и так далее, пока оба подыгрывающих не примут положительное решение и не прекратят свою работу. Таким образом, безвредный противник играет роль проводника, связывающего между собой пользователей *i* и *j*.

В заключение описания сеансов протокола примем, что *t*-й запрос нарушителя посылается подыгрывающему в момент времени $t = t_t \in R$, и потребуем, чтобы при *t* < *u* выполнялось неравенство $t_t < t_u$.

5.2.4. Формальное определение стойкости

Как следует из описания протокола, целью взаимной аутентификации пользователей являются согласованные диалоги. Тогда понятие стойкости протокола взаимной аутентификации можно сформулировать следующим образом.

Определение 1. *Протокол* $\prod(\mathrm{k}, \{\mathrm{A}, \mathrm{B}\})$ *является стойким протоколом аутентификации, выполняемым пользователями A и B, если оба подыгрывающих* $\prod_{A,B}^{s}$ *и* $\prod_{B,A}^{t}$ *принимают положительное решение тогда и только тогда, когда они ведут согласованные диалоги, причем вероятность противоположного события пренебрежимо мала.*

Диалог подыгрывающего $\prod_{i,j}^{s}$ представляет собой последовательность упорядоченных во времени сообщений, отсылаемых им в сеть, и получаемых на них ответов. Пусть $t_1 < t_2 < \ldots < t_R$ (где *R* – некоторое положительное целое число) – моменты времени, в которые подыгрывающий $\prod_{i,j}^{s}$ посылает сообщение. Диалог можно обозначить следующим образом:

$$conv = \left(t_1, m_1, m_1^{'}\right), \left(t_2, m_2, m_2^{'}\right), \ldots, \left(t_R, m_R, m_R^{'}\right).$$

Эта запись означает, что в момент t_1 подыгрывающий $\prod_{i,j}^{s}$ получает запрос m_1 и посылает в ответ сообщение $m_1^{'}$. Затем в момент $t_2 > t_1$ подыгрывающий получает запрос m_2 и посылает в ответ сообщение $m_2^{'}$ и так далее, пока в момент t_R он не получит запрос m_R и пошлет в ответ сообщение $m_R^{'}$.

Удобно предположить, что диалог начинает нарушитель. Итак, если m_1 = "", будем называть подыгрывающего $\prod_{i,j}^{s}$ инициатором диалога, в противном случае – ответчиком.

Пусть $conv = (t_0, "", m_1), (t_2, m_1', m_2), (t_4, m_2', m_3), ..., (t_{2t-2}, m_{t-1}', m_t)$ обозначает диалог подыгрывающего $\prod_{i,j}^{s}$. Будем говорить, что подыгрывающий $\prod_{j,i}^{t}$ ведет диалог *conv'*, согласованный с диалогом *conv*, если существует последовательность моментов времени $t_0 < t_1 < t_2 < ... < t_R$, такая что

$$conv = (t_1, m_1, m_1'), (t_3, m_1, m_2'), (t_5, m_3, m_3'), ..., (t_{2t-2}, m_t, m_t'),$$

где строка m_t' означает «нет сообщений».

Если подыгрывающие $\prod_{i,j}^{s}$ и $\prod_{j,i}^{t}$ всегда ведут согласованные диалоги, то нарушитель не в состоянии организовать опасную атаку и вынужден играть роль безвредного противника.

Если протокол стойкий, то из существования согласованных диалогов непосредственно следуют положительные решения подыгрывающих. Доказать обратное утверждение, т.е. что из положительных решений следует существование согласованных диалогов, намного сложнее. Следовательно, целью атаки на протокол является получение положительных решений, когда согласованных диалогов не существует. Таким образом, более корректным является следующее определение стойкости протокола.

Определение 2. *Протокол* $\prod(k, \{A, B\})$ *является стойким протоколом аутентификации, выполняемым пользователями A и B, если вероятность выигрыша нарушителя, заключающегося в том, что оба подыгрывающих* $\prod_{A,B}^{s}$ *и* $\prod_{B,A}^{t}$ *принимают положительное решение при отсутствии между ними согласованных диалогов, пренебрежимо мала.*

5.2.5. Формальное доказательство стойкости

Рассмотрим формальное доказательство стойкости простого протокола взаимной аутентификации ППВА.

Исходные условия

Абоненты (*A*) (*B*) обладают общим секретным симметричным ключом *K*, имеющим размер *k*;

R_A – одноразовое случайное число абонента A;

R_B – одноразовое случайное число абонента B, причем оба числа имеют длину *k*;

$[x]_K$ – пара (x, псф$_K$(x)), где $x \in \{0,1\}^*$;

Псф$_K$: $\{0,1\}^* \rightarrow \{0,1\}^k$ – псевдослучайная функция, использующая ключ *K*.

Описание протокола

Протокол начинается с того, что абонент А посылает абоненту В сообщение $A||R_A$, где R_A – случайное число абонента А, имеющее длину k. Абонент В отвечает, генерируя случайный оклик R_B длиной k и отсылая обратно сообщение $[B \parallel A \parallel R_A \parallel R_B]_K$. Если сообщение имеет правильный формат и метку, абонент А отсылает абоненту В сообщение $[A||R_B]_K$ и аутентифицирует его. Затем абонент В проверяет корректность сообщения абонента А. Если оно имеет правильный формат и метку, абонент В, в свою очередь, аутентифицирует абонента А.

Если между абонентами А и В вклинивается безвредный противник, то в моменты времени $t_0 < t_1 < t_2 < ... < t_3$ абонент А принимает диалоги

$$conv_A = (t_0, "", A \parallel R_A), (t_2, [B \parallel A \parallel R_A \parallel R_B]_K, [A \parallel R_B]_K),$$

а абонент В – диалоги

$$conv_B = \left(t_1, A \parallel R_A, \ [B \parallel A \parallel R_A \parallel R_B]_K\right), \ \left(t_3 [A \parallel R_B]_K, \text{ «сообщений нет»}\right).$$

Очевидно, что диалоги $conv_A$ и $conv_B$ являются согласованными.

Доказательство

Для доказательства стойкости протокола ППВА к любой полиномиально ограниченной атаке анализируется два эксперимента. В первом эксперименте функция Псф_K является абсолютно случайной. Иначе говоря, подыгрывающие $\text{ППВА}^{s}_{A,B}$ и $\text{ППВА}^{t}_{B,A}$ одновременно распоряжаются функцией Псф_K. Когда они применяют ее к аргументу x, результатом $\text{Псф}_K(x)$ является строка, равномерно распределенная по множеству $\{0,1\}^k$. Необходимо отметить, что способ реализации такой функции пока неизвестен. Во втором эксперименте протокол ППВА реализуется с помощью семейства псевдослучайных функций, которые применяются на практике.

Поскольку в первом эксперименте результат $\text{Псф}_K(x)$ является k-битовой равномерно распределенной строкой, подыгрывающий $\text{ППВА}^{s}_{A,B}$ видит диалог $conv_A$ и убеждается, что равномерно распределенная строка $[B \parallel A \parallel R_A \parallel R_B]_K$ вычислена с помощью значения R_A, сгенерированного им самим. Значит, вероятность того, что эта строка была вычислена нарушителем, пренебрежимо мала и равна 2^{-k}. Эта величина не зависит от возможностей нарушителя. Итак, подыгрывающий $\text{ППВА}^{s}_{A,B}$ считает, что требуемый партнер видит диалог $\left(t_1, A \parallel R_A, [B \parallel A \parallel R_A \parallel R_B]_K\right)$. По сути, это доказывает, что существует диалог, согласованный с диалогом $conv_A$ и вычисленный партнером подыгрывающего $\text{ППВА}^{s}_{A,B}$ с огромной вероятностью, зависящей от величины k.

Аналогично, когда подыгрывающий $\text{ППВА}^{t}_{B,A}$ видит диалог $conv_B$, он убеждается, что согласованный диалог создан его партнером с той же вероятностью, зависящей от величины k.

Таким образом, если протокол ППВА реализуется с помощью истинно случайной функции, общей для обоих подыгрывающих, вероятность выигрыша нарушителя является пренебрежимо малой относительно величины k.

Оставшаяся часть доказательства сводится к обнаружению противоречия.

Предположим, что во втором эксперименте нарушитель выигрывает со значимой вероятностью, зависящей от параметра k. Построим полиномиальный алгоритм распознавания T, отличающий случайные функции от псевдослучайных. Алгоритм T получает на вход функцию g:$\{0,1\}^* \rightarrow \{0,1\}^k$, выбранную в ходе следующего эксперимента.

1. Подбрасываем идеальную монету C.
2. Если C = «ОРЕЛ», функция g считается случайной.
3. Иначе генерируем случайное число K и полагаем g – Псф$_K$.

Цель алгоритма T – предсказать результат жеребьевки со значимой вероятностью. Его стратегия заключается в том, чтобы выполнить атаку на протокол ППВА, реализованный с помощью функции g.

Если нарушитель выигрывает, то алгоритм T выдает результат C – «ОРЕЛ» (т.е. функция g является псевдослучайной), в противном случае алгоритм T возвращает результат C = «РЕШКА» (т.е. функция g является абсолютно случайной). Итак, преимущество, с которым алгоритм T отличает k-битовые случайные и псевдослучайные функции, равно преимуществу, с которым нарушитель может выиграть, т.е. является значимым. Это противоречит общепринятому убеждению, что не существует полиномиального алгоритма распознавания, позволяющего различать случайные и псевдослучайные функции. На практике псевдослучайную функцию Псф$_K$ можно реализовать с помощью функции формирования имитоставки или функции хэширования HMAC. Обе эти реализации являются эффективными.

Аналогичным образом можно доказать стойкость протоколов согласования сеансового ключа, протоколов обмена ключами и протоколов аутентификации с участием третьей доверенной стороны.

5.3. Доказательство корректности протоколов с помощью логических правил

В этих методах стойкость демонстрируется с помощью операций над абстрактными символами. В одних работах для анализа стойкости используются формальные логические системы, в других – автоматизированные средства для доказательства математических теорем, возвращающие результат ДА/НЕТ.

Подход, основанный на доказательстве теорем, состоит в следующем.

1. Определяется множество алгебраических или логических символов, используемых для описания поведения системы либо утверждений, которые могут быть как предпосылками (известными формулами), так и следствиями (результатами вывода).

2. Постулируется набор аксиом, устанавливающих правила вывода формул.

3. Поведение системы описывается в виде набора доказываемых теорем.

4. При доказательстве теорем используются предпосылки и аксиомы, а также ранее доказанные теоремы.

Иногда процесс доказательства можно автоматизировать с помощью определенных правил применения аксиом и теорем. Как правило, для автоматизации доказательства применяются правила подстановки термов. Например, широко известно, что любое булево выражение можно автоматически переписать в «конъюнктивной нормальной форме». Однако в большинстве случаев автоматизированное доказательство теорем является слишком долгим и трудоёмким. Хорошо известно, что длина автоматизированного доказательства может оказаться неполиномиально ограниченной величиной, зависящей от размера доказываемой формулы, т.е. практически неразрешимой задачей.

Несмотря на то что автоматизированные доказательства теорем часто оказываются слишком длинными, этот метод нашел широкое применение при исследовании систем, поведение которых невозможно описать с помощью конечной структуры (например, системы с бесконечным пространством состояний). Примером такого доказательства является метод математической индукции. Однако для получения короткого доказательства необходимо вмешательство человека.

Необходимым свойством алгебраического доказательства теоремы является аксиоматическая система, сохраняющая так называемое свойство *конгруэнтности*. Это свойство является обобщением понятия сравнимости между целыми числами на произвольные алгебраические структуры. Бинарное отношение R, определенное в алгебраической структуре, называется конгруэнтностью, если для любой бинарной операции $\circ$ над элементами этой структуры из условий

$$R(x, u) \text{ и } R(y, v)$$

следует утверждение

$$R(x \circ y, u \circ v).$$

Конгруэнтность иногда называют также свойством подстановки, или замены. В таком случае один компонент системы можно заменить другим, не нарушая согласованности между элементами системы, иначе автоматизированное доказательство теорем становится необоснованным. По этой причине свойство подстановки часто называют свойством семантической непротиворечивости системы для доказательства теорем. Семантически противоречивое доказательство теорем лишено смысла, поскольку может привести к ложному утверждению, например, к абсурдному выводу «1 = 2».

Полнота системы для доказательства теорем зависит от того, обладает ли выбранная система аксиом свойством семантической истинности. Если система для доказательства теорем является полной, любое семантически истинное утверждение должно быть доказуемым, т.е. должна существовать последовательность применения аксиом, демонстрирующая синтаксическую истинность утверждения. Полнота системы желательна, но, как правило, автоматизированные системы доказательства теорем ею не обладают.

Целью доказательства теоремы является демонстрация требуемых свойств системы или обнаружение ошибок. Именно поэтому нежелательное свойство нельзя сформулировать в виде теоремы. Несмотря на это, невозможность доказать требуемое свойство при доказательстве теоремы часто приводит к выявлению скрытых ошибок.

Протоколы аутентификации представляют собой чрезвычайно уязвимые системы. Как правило, невозможно сначала разработать неуязвимый протокол, а потом формально доказать его стойкость с помощью доказательства теоремы. Следовательно, этот подход необходимо применять уже на этапе разработки протокола.

5.4. Методы доказательства, основанные на исследовании состояний системы

Другой подход к формальному анализу сложных систем основан на моделировании поведения конечных систем. В этом случае состояния системы выражаются с помощью определенных отношений. Анализ поведения системы, как правило, сводится к исследованию пространства состояний и проверке заданных отношений. Эта методология называется проверкой моделей.

Проверка моделей должна гарантировать, что нежелательные состояния никогда не возникнут, либо в конце концов система будет находиться в требуемом состоянии.

Проверка модели сводится к следующим действиям.

1. Функционирование конечной системы моделируется с помощью конечной системы переходов (КСП), переводящей систему из одного состояния в другое в зависимости от происходящих событий.

2. Каждое состояние системы КСП выражается с помощью логической формулы.

3. Требуемое свойство системы также явно выражается логической формулой.

4. Осуществляется символическое выполнение системы КСП, которое описывается трассой

$$p = f_0 e_1 f_1 e_2 \ldots e_n f_n,$$

где $f_0, \ldots, f_n$ – логические формулы, а $e_1, \ldots, e_n$ – события.

5. Используется автоматическая процедура проверки, является ли конечная целевая формула логическим следствием трассы.

Следует отметить, что в отличие от доказательства теорем, описывающих только желательные свойства, при проверке модели результирующая формула может выражать как желательное, так и нежелательное состояние системы. Например, утверждение «нарушителю известен новый сеансовый ключ K» представляет собой формулу, выражающую нежелательное свойство протокола распределения сеансовых ключей. Если результирующая формула выражает

нежелательное свойство, то проверка модели создает трассу, приводящую к явному описанию системной ошибки. Таким образом, проверку моделей можно применять для выявления ошибок, скрытых в системе. Как правило, проверку моделей при анализе протоколов аутентификации применяют именно для этой цели.

При разработке сложных систем их, как правило, конструируют из более простых компонентов. Методом системной композиции можно упростить выявление дефектов в протоколах аутентификации с помощью проверки моделей. Действительно, поиск ошибок в протоколах аутентификации представляет собой процесс проверки системы, которая всегда крупнее, чем системы, имитирующие отдельные части протокола. В лучшем случае спецификация протокола только описывает законные действия пользователей. Однако удачная атака всегда описывает поведение более крупной системы, в которой нарушитель согласовывает свое поведение с законными пользователями (т.е. обманывает их или незаметно раскрывает секреты).

Следовательно, при анализе протоколов аутентификации с помощью проверки модели необходимо моделировать не только действия законных пользователей, но и типичное поведение нарушителя. Каждый из этих компонентов протокола является системой КСП. Эти компоненты объединяются в более сложные системы, а затем подвергаются проверке. Операция композиции в ходе проверки моделей часто моделирует асинхронную связь между компонентами системы. Термин «асинхронный» означает, что составная система может выполнять операции в зависимости от действий подсистем и описывает ситуацию, в которой поведение нарушителя может не зависеть от действий законных пользователей.

Так как проверка моделей предназначена для анализа конечных систем, то при анализе протоколов аутентификации нарушитель должен быть полиномиально ограниченным – действия, связанные с неограниченной вычислительной мощностью, не рассматриваются.

При проверке моделей часто возникает проблема «комбинаторного взрыва»: система отображается в крупную систему КСП, имеющую слишком большое количество состояний, при этом для их обработки имеющихся вычислительных ресурсов недостаточно. Эта проблема имеет особенно острый характер при анализе крупного программного или аппаратного обеспечения. Такие системы имеют огромное пространство состояний. Однако при разумном предположении, что нарушитель является полиномиально ограниченным, большинство протоколов аутентификации можно смоделировать с помощью относительно небольших КСП-систем.

При анализе протоколов аутентификации применяются два способа проверки моделей: анализаторы протоколов и алгебра процессов.

5.4.1. Анализаторы протоколов

Работа анализаторов протоколов основана на предположении об ограниченности вычислительных возможностей нарушителя. Несмотря на то

что нарушитель, выдавая себя за законного пользователя, может контролировать обмен сообщениями в сети (перехватывать их, читать, модифицировать или уничтожать, выполнять преобразование перехваченных сообщений и пересылать их другим участникам протокола), существует определенное множество «слов», которые остаются неизвестными как в начале выполнения протокола, так и после его завершения, если протокол оказался стойким. Эти слова могут быть секретными сообщениями или криптографическими ключами, которые защищаются протоколом. Назовем эти слова «ключевыми». Если целью нарушителя является распознавание ключевых слов, задача доказательства стойкости протокола формулируется следующим образом: ключевые слова должны остаться неизвестными. И наоборот, доказательство уязвимости протокола означает, что в итоге ключевые слова становятся известными нарушителю.

В анализаторе протоколы, как правило, моделируются с помощью глобальной конечной системы. Эта система состоит из нескольких локальных подсистем и содержит определенную информацию о состояниях, доступную для нарушителя. Каждая локальная подсистема описывает функционирование истинного участника протокола. Такой способ конструирования системы следует стандартной методологии создания сложных систем из более простых компонентов.

Включение нарушителя в глобальную систему моделирует способ, которым он получает информацию в результате выполнения протокола. Целью нарушителя является генерация «ключевых» слов путем перевода истинных пользователей в определенное нежелательное состояние, противоречащее протоколу. Такое состояние называется «критическим». Если протокол содержит ошибку, пользователь может оказаться в критическом состоянии. В модели перезаписи термов критическое состояние эквивалентно возникновению последовательности термов, демонстрирующей, что некоторое слово, которое должно быть недоступным нарушителю («ключ»), становится ему известным.

В анализаторе протокола определяется и набор правил перехода из одного состояния в другое. При определенных условиях правило перехода может «сработать». Для этого должны выполняться два условия.

1. Нарушителю приписываются определенные слова.
2. Соответствующие локальные состояния оказываются связанными с определенными значениями.

После применения правила происходит следующее.

1. Истинный пользователь выводит определенные слова, которые таким образом становятся известными нарушителю.
2. Соответствующие локальные состояния оказываются связанными с новыми значениями.

Слова, связанные с правилом, подчиняются правилам перезаписи термов. Обычно в системе существуют три правила, выражающие понятие равенства, а

также тот факт, что зашифрование и расшифрование являются обратными функциями. Эти правила выглядят следующим образом:

encrypt (X, decrypt(X, Y)) $\rightarrow Y$

decrypt (X, encrypt(X, Y)) $\rightarrow Y$

id_check(X, X) $\rightarrow$ ДА

Чтобы выполнить анализ, пользователь посылает анализатору запрос о состоянии с помощью слов, известных нарушителю (т.е. описание критического состояния). Затем анализатор протокола выполняет обратный поиск, пытаясь обнаружить исходное состояние глобальной системы. Для этого можно попытаться применить к текущему состоянию правую часть правила перезаписи термов и свести ее к левой части, которая описывает предыдущее состояние. Если исходное состояние обнаружено, система действительно является нестойкой, в противном случае предпринимается попытка доказать, что исходное состояние недостижимо. Для этого необходимо показать, что любое состояние, из которого можно перейти в указанное состояние, также является недостижимым. Эта разновидность поиска чревата возникновением бесконечного следа, в котором для того, чтобы нарушитель распознал слово A, необходимо, чтобы он узнал слово B, а для этого он должен узнать слово C, и т. д. Следовательно, анализатор должен обладать определенными возможностями, позволяющими пользователю доказать лемму о недостижимости определенного класса состояний. Цель этой процедуры – уменьшить пространство состояний так, чтобы стал возможным исчерпывающий поиск, позволяющий определить стойкость протокола. Хорошо известно, что такой алгоритм может оказаться бесконечным. Следовательно, необходимо ограничить количество рекурсивных вызовов некоторых проверяющих процедур. Работа с анализатором требует от пользователя большой аккуратности при кодировании правил перехода и описании критического состояния.

Анализаторы протоколов успешно используются для проверки большого количества протоколов аутентификации. В частности, с их помощью были проанализированы протоколы аутентификации с открытым ключом, протокол IKE и протоколы безопасных электронных транзакций SET (Secure Electronic Transaction).

5.4.2. Алгебра процессов

Алгеброй процессов (системой алгебры процессов – САП) называют целое семейство систем, основанных на алгебраическом подходе к построению абстрактных вычислительных структур. Поскольку САП – это алгебра, она представляет собой язык термов, для которых определены операции. Эти операции подчиняются аксиоме замкнутости, т.е. термы САП образуют множество, замкнутое относительно указанных операций. Каждая операция имеет определенный смысл, согласованный со смыслом термов.

5.4.2.1. Действия, события и процессы в САП

В алгебре САП система описывается в терминах действий, которые она может выполнять. Действием называется конечная последовательность возникающих друг за другом событий. В частности, действие может описываться последовательностью нулевой длины, которая означает отсутствие каких-либо действий. Множество всех возможных событий (зафиксированных перед началом анализа) называется *алфавитом процесса* и обозначается символом $\sum$. Следовательно, для любого действия выполняется условие $a \in \sum^*$. Например, алфавитом процесса может служить множество $\sum = \{0,1\}$, а действием, выполняемым таким процессом, может являться битовая строка, обладающая определенным свойством.

При моделировании протоколов или систем связи с помощью алгебры САПР действие представляет собой отдельное сообщение или последовательность сообщений. Если M и N – последовательности сообщений, то $M.N$ – также последовательность сообщений (иногда при записи последовательности сообщений символ «точка» между ними пропускается).

Процессы являются компонентами системы. Они представляют собой сущности, описываемые алгеброй САП в терминах возможных действий, которые они могут выполнять. Основные процессы алгебры САП и их смысловое содержание включают в себя:

– *STOP* (отсутствие действий) – бездействие;

– $a \rightarrow P$ (префикс) – сущность выполняет действие a, а затем функционирует, как пользователь P;

– $P \square Q$ (детерминированный выбор) – сущность ведет себя как пользователь P или Q в зависимости от внешнего события;

– $P \wedge Q$ (недетерминированный выбор) – сущность ведет себя как пользователь P или Q по непонятным причинам;

– $/\, a$ (сокрытие) – не обращать внимание на действие a;

– $mX.P(x)$ (рекурсия) – повторить действия пользователя P над переменной X, т.е. $mX.P(x) \overset{\text{def}}{=} P(mX.P(x))$;

– $P \parallel Q$ (параллельность) – пользователи P и Q одновременно выполняют одно и то же действие;

– $P \,|||\, Q$ (чередование) – пользователи P и Q не обязаны выполнять одно и то же действие.

Перечисленные выше основные операции представляют собой строительные конструкции процессов в алгебре САП, предназначенные для моделирования и описания функционирования конечной системы. Эти конструкции и семантика операций делают язык САП достаточно мощным для того, чтобы описывать сложные конечные системы.

Процесс P представляет собой множество последовательностей возможных событий – tr(P) (трасс). К этому множеству относится также пустая трасса {}.

Операция «.» над двумя множествами трасс T и T' определена следующим образом:

$$T.T' = \{tr.tr' \mid tr \mid \in T \wedge tr' \in T'\},$$

где операция конкатенации посредством точки определена выше.

Процесс *P* соответствует языку *L* (например, его спецификации), если все его трассы являются частью языка *L*:

$$\mathrm{P\,sat\,L} \Leftrightarrow \mathrm{tr(P)} \subseteq \mathrm{L}.$$

Процесс *P* уточняет процесс *Q*, если $\mathrm{tr(P)} \subseteq \mathrm{tr(Q)}$. Отсюда, если процесс *Q* соответствует языку *L* (т.е. *Q sat L*), то процесс *P* также соответствует ему.

Подход, основанный на проверке моделей, позволяет автоматически проверять отношения между конечными процессами, используя метод уточнения расхождений FDR, разработанный компанией Formal Systems (Europe) Ltd.

Выше отмечалось, что композиция систем играет существенную роль в проверке моделей. В алгебре САП композиция систем достигается с помощью операций «параллельность» и «чередование».

Применяя «аксиому блокировки», композицию систем можно создать автоматически, заменяя операцию чередования операцией параллельность каждый раз, когда объединяемые термы обозначают одно и то же действие.

5.4.2.2. Анализ стойкости протоколов

Операция композиции в алгебре САП делает ее особенно полезной для моделирования и описания поведения параллельных систем и средств связи. Именно это обстоятельство побудило некоторых исследователей утверждать, что алгебра САП позволяет анализировать протоколы аутентификации. Кроме того, существует модель FDR, которая используется в сочетании с алгеброй САП для уточнения процессов. С помощью этой модели при анализе конфиденциальности сообщений используется ещё одно отношение $I \vdash m$ (следование), которое позволяет указать способ извлечения информации из доступных данных. Ниже представлены аксиомы следования, описывающие извлечение *I* – исходной информации:

– если $m \in I$, то $I \vdash m$;
– если $I \vdash m$ и $I \subseteq I'$, то $I' \vdash m$;
– если $I \vdash m_1$, и $I \vdash m_2$, то $I' \vdash (m_1, m_2)$ (спаривание);
– если $I \vdash (m_1, m_2)$ то $I \vdash m_1$ и $I \vdash m_2$ (проекция);
– если $I \vdash m$ и $I \vdash K \in K$, то и $I \vdash \{m\}_K$ (зашифрование);
– если $I \vdash \{m\}_K$ и $I \vdash K^{-1} \in K$, то и $I \vdash m$ (расшифрование).

Например, если

$$(\{\{K_1\}_{K2}\}_{K3}, K_3) \vdash K_3$$

и

$$(\{\{K_1\}_{K2}\}_{K3}, K_3) \vdash \{\mathrm{K_1}\}_{\mathrm{K_2}},$$

то

$$(\{\{K_1\}_{K2}\}_{K3}, K_3) \vdash \mathrm{K_1}\ (\text{ложь}).$$

Аксиомы следования интуитивно описывают, как именно нарушитель извлекает информацию. Пытаясь скрытно взломать протокол, нарушитель может использовать исходную информацию, которой он владеет, и некоторые протокольные сообщения, перехваченные в сети. Хотя по определению множество I является бесконечным, на практике для данного протокола множество I можно сузить до конечного множества полезной информации. Более того, на самом деле не обязательно конструировать множество I. Достаточно проверить условие $m \in I$ для некоторого конечного набора сообщений.

Вышеизложенное показывает, что применение автоматизированных средств в рамках алгебры САП является важным элементом анализа функционирования систем. При конструировании системы из небольших компонентов с помощью операции композиции по определённым выше семантическим правилам можно создать более крупные системы. В процессе уточнения процессов автоматизированные средства также позволяют применить процедуру уточнения и аксиомы следования. Синтаксис САП, учитывающий линии связи между компонентами системы, не очень удобен для человека, зато вполне пригоден для реализации с помощью автоматизированных средств.

Контрольные вопросы и задачи

1. Назовите наиболее распространённые атаки на протоколы аутентификации.
2. Какая модель нарушителя используется при оценке стойкости протоколов аутентификации?
3. Охарактеризуйте методы формальной оценки протоколов аутентификации.
4. Неправильное применение криптографических средств – распространенная ошибка, возникающая при разработке протоколов аутентификации. В чем именно проявляется эта ошибка?
5. Назовите этапы формального доказательства стойкости в рамках вычислительной модели.
6. В чём заключается подход формальной оценки стойкости, основанный на доказательстве теорем?
7. В каких случаях при анализе протоколов аутентификации используются анализаторы протоколов и алгебра процессов?

ЛИТЕРАТУРА

1. Романец, Ю. В. Защита информации в компьютерных системах и сетях / Ю. В. Романец, П. А. Тимофеев, В. Ф. Шаньгин. – М.: Радио и связь, 2001. – 376 с.

2. Конявский, В. А. Управление защитой информации на базе СЗИ НСД «Аккорд» / В. А. Конявский. – М.: Радио и связь, 1999. – 325 с.

3. Мао, Венбо. Современная криптография: теория и практика / Венбо Мао; пер. с англ. – М.: Издательский дом «Вильямс», 2005. – 768 с.

4. Кухарев, Г. А. Биометрические системы. Методы и средства идентификации личности человека / Г. А. Кухарев. – СПб.: Политехника, 2001. – 240 с.

5. Бобов, М. Н. Обеспечение безопасности информации в телекоммуникационных системах / М. Н. Бобов, В. К. Конопелько. – Минск : БГУИР, 2002. – 164 с.

6. Вычислительные сети и сетевые протоколы / Д. Дэвис [и др.]. – М.: Мир, 1982. – 562 с.

7. Бабенко, Л. К. Защита информации с использованием смарт-карт и электронных брелоков / Л. К. Бабенко, С. С. Ищуков, О. Б. Макаревич. – М.: Гелиос АРВ, 2003. – 352 с.

8. Дшхуннян, В. Л. Электронная идентификация. Бесконтактные электронные идентификаторы и смарт-карты / В. Л. Дшхуннян. – М.: ООО «Издательство АСТ», Издательство «НТ Пресс», 2004. – 695 с.

9. Пластиковые карты / А. А. Андреев [и др.]. – М.: БДЦ-Пресс, 2002. – 576 с.

10. Смит, Ричард Э. Аутентификация: от паролей до открытых ключей / Ричард Э. Смит. – М.: Издательский дом «Вильямс», 2002. – 432 с.

11. Хоффман, Л. Современные методы защиты информации / Л. Хоффман. – М.: Сов. радио, 1980. – 264 с.

12. Саломаа, А. Криптография с открытым ключом / А. Саломаа; пер. с англ. – М. : Мир, 1995. – 304 с.

13. Рабинер, Л. Р. Цифровая обработка речевых сигналов / Л. Р. Рабинер, Р. В. Шафер : пер. с англ. – М.: Радио и связь, 1981. – 495 с.

14. Трахтенброт, Б. А. Конечные автоматы (поведение и синтез) / Б. А. Трахтенброт, Я. М. Барздинь. – М.: Наука, 1970. – 400 с.

15. Оре, О. Теория графов / О. Оре. – 2-е изд. – М.: Наука, 1980. – 336 с.

16. Мафтик, С. Механизмы защиты в сетях ЭВМ / С. Мафтик; пер. с англ. – М.: Мир, 1993. – 216 с.

17. Грушо, А. Аутентификация и авторизация: новый взгляд / А. Грушо, А. Балакин, А. Сарубков // Connect. – 2004. – №7. – С. 108 – 110.

18. Эйкхофф, П. Основы идентификации систем управления / П. Эйкхофф. – М.: Мир, 1975. – 684 с.

19. Jianxin, Yan. Memorability and Security of Passwords – Some Empirical Results / Yan Jianxin [and other] – Cambridge, UK: Cambridge University Computer Laboratory, 2001. – 227 p.

20. Anil, Jain. Biometrics: Personal Identification in Networked Society / Jain Anil, Ruud Bolle, Sharath Pankanti. – Boston: Kluwer Academic Publishers, 1999. – 356 p.

21. Иванов, А. И. Оценка эффекта от использования тайных биометрических образов / А. И. Иванов // Защита информации. Конфидент – 2002. – №4 – 5. – С. 128 – 131.

22. Введение в биометрию. Ч. 4. Идентификация по отпечаткам пальцев – наиболее развитая биометрическая технология // «БДИ» – 2005. – №2. – С. 36 –38.

23. Введение в биометрию. Ч. 5. Биометрическая идентификация – это не только отпечатки пальцев… // «БДИ» – 2005. – №3. – С. 48 – 52.

24. Feldmeier, David C. UNIX Password Security – Ten Years Later / David C. Feldmeier, Philip R. Karn – Heidelberg: Springer-Verlag, 1990. – 479 p.

25. Техническое описание контроллера для электронных идентификаторов iButton STM-8L. [Электронный ресурс]. Режим доступа: http://www.digitalas.lt/files/1127798850.pdf – 04.04.2007.

26. Петрикович, Я. Я. RU 2212053 C1 МПК G06 K9/00, A61 B5/117. Датчик изображения папиллярных линий кожи пальца / Я. Я. Петрикович, И. А. Кан, С. Т. Иванченко. Дата заяв. 06.11.2002, дата публ. 10.09.2003.

27. Медль, А. RU 2267159 C2 МПК G06 K9/46. Способ и система для генерирования набора параметров ключа доступа, а также для аутентификации человека на основании его биометрического параметра / А. Медль, Э. Штефан, Р. Мюллер. Дата заяв. 07.05.2001, дата публ. 15.11.2001.

28. RU 2161336 C2 МПК G10 L17/00. Система для верификации говорящего / Ричард Дж. Мэммон, К. Феррел [и др.]. Дата заяв. 07.06.1995, дата публ. 27.12.2000.

29. RU 2093890 C1 МПК G06 K9/00, G07 C 9/00. Способ распознавания личности и система для его осуществления / О. А. Серебренников, А. Б. Мурынин [и др.]. Дата заяв. 08.09.1995, дата публ. 20.10.1997.

30. Шаров, В. Биометрические методы компьютерной безопасности / В. Шаров // Byte Россия. – 2005. – С. 23 – 29.

31. Бобов, М. Н. Методы оценки показателей защищенности автоматизированных систем от несанкционированного доступа / М. Н. Бобов, А. А. Обухович // Методы и технические средства обеспечения безопасности информации: тез. докл. науч.-техн. конференции. – СПб., 1997. – С. 19 – 20.

32. Бобов, М. Н. Оценка уровня защищенности средства аутентификации по отпечатку пальца / М. Н. Бобов, П. М. Буй // Управление защитой информации. – 2008.– №1. – С. 31 – 37.

33. Рылов, А. С. Анализ речи в распознающих системах / А. С. Рылов. – Минск : Бестпринт, 2003. – 264 с.

34. Бобов, М. Н. Оценка уровня защищенности голосового средства аутентификации / М. Н. Бобов, П. М. Буй // Информатика. – 2008. – №1. – С. 31–37.

35. Иванов, В. П. К вопросу о выборе технических средств защиты информации от НСД / В. П. Иванов, А. В. Иванов // Защита информации. INSIDE. Ч. 1. – 2006. – №1. – С. 48 – 54, Ч. 2. – 2006. – №2. – С. 62 – 67.

36. Бобов, М. Н. Синтез модели средства аутентификации / М. Н. Бобов, П. М. Буй // Доклады БГУИР. – 2007. – №5. – С. 23 – 31.

37. Буй, П. М. Исследование модели средств аутентификации / П. М. Буй // Доклады БГУИР. – 2007. – №5. – С. 32 – 38.

38. Диффи, У. Защищенность и имитостойкость. Введение в криптографию / У. Диффи, М. Э. Хеллмэн // ТИЭР. – 1979. – Т.67. – №3. – С. 71 – 100.

39. Симонс, Г. Дж. Обзор методов аутентификации информации / Г. Дж. Симонс // ТИИЭИР. – 1988. – Т.76. – №5. – С. 105 – 125.

Св. план 2008, поз. 100

Учебное издание

Бобов Михаил Никитич
Конопелько Валерий Константинович

ОСНОВЫ АУТЕНТИФИКАЦИИ В ТЕЛЕКОММУНИКАЦИОННЫХ СИСТЕМАХ

Пособие по дисциплинам
«Защита программного обеспечения и баз данных в сетях телекоммуникаций» и «Биометрические системы контроля доступа в сетях телекоммуникаций» для студентов телекоммуникационных специальностей 1-45 01 05 и 1-98 01 02 всех форм обучения

Редактор Г. С. Корбут
Корректор Е. Н. Батурчик
Компьютерная верстка Е. Г. Бабичева

Подписано в печать 23.10.2009. Формат 60x84 1/16. Бумага офсетная.
Гарнитура «Таймс». Печать ризографическая. Усл. печ. л. 7,91.
Уч.-изд. л. 8,0. Тираж 150 экз. Заказ 25.

Издатель и полиграфическое исполнение: Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
ЛИ №02330/0494371 от 16.03.2009. ЛП №02330/0494175 от 03.04.2009.
220013, Минск, П. Бровки, 6