МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ АГЕНТСТВО ПО ОБРАЗОВАНИЮ

Государственное образовательное учреждение
высшего профессионального образования
"Оренбургский государственный университет"

Т.Н. ШАЛКИНА

МЕТОДЫ И СРЕДСТВА ЗАЩИТЫ ИНФОРМАЦИИ В ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМАХ И СЕТЯХ

Рекомендовано Ученым советом государственного образовательного учреждения высшего профессионального образования "Оренбургский государственный университет" в качестве учебного пособия для студентов, обучающихся по программам высшего профессионального образования по направлению 230100 «Информатика и вычислительная техника»

Оренбург 2007

УДК 004.056.5(076.5)
ББК 32.973-018.2я73
Ш 20

Рецензент
доктор технических наук, профессор Т.З. Аралбаев

Шалкина Т.Н.

Ш 18 **Методы и средства защиты информации в вычислительных системах и сетях: учебное пособие / Т.Н. Шалкина - Оренбург, ГОУ ОГУ, 2007. – 245 с.**

ISBN

В учебном пособии рассматриваются современные методы и средства, используемые для защиты данных в вычислительных системах и сетях. Пособие содержит методические материалы для проведения лабораторного практикума по дисциплине «Методы и средства защиты компьютерной информации».

Пособие предназначено для студентов специальностей направления «Информатика и вычислительная техника», может быть полезно преподавателям и аспирантам соответствующего профиля.

ББК 32.973-018.2я73

2404090000

Ш ———————

ISBN

Содержание

1 Безопасность вычислительной системы

1.1 Основные понятия и определения

В настоящее время мировое сообщество переживает новый этап своего развития – стадию построения информационного общества, для которого знания и информация являются основными ценностями. Объем научно-технической информации, необходимый для поддержания технологических процессов производства, резко возрастает; устраняются ограничения в области накопления и использования информационных ресурсов во всех сферах социальной практики. Увеличение компьютерной техники и средств связи для обработки, передачи и хранения информации во всех сферах человеческой жизнедеятельности актуализирует проблему обеспечения защиты и сохранности информации.

Анализ публикаций по проблеме защиты информационных ресурсов показывает необходимость разработки и использования специальных методов и средств для защиты информации на всех уровнях: начиная от персонального компьютера и заканчивая вычислительными сетями. Особенно актуальными становятся проблемы защиты информации в глобальной вычислительной сети Интернет, что обусловлено широким использованием последней в электронной коммерции, обучении и т.п. В данном пособии рассматриваются современные методы и средства, используемые для защиты информации в вычислительных системах и сетях. Особое внимание в работе уделено способам защиты Интернет-ресурсов.

Введем основные термины, которые будем использовать в дальнейшем. [9-11, 14, 30, 33]

Под ***безопасностью информации*** понимают состояние защищенности информации, обрабатываемой средствами вычислительной техники или автоматизированной системы, от внутренних или внешних угроз. В свою очередь, защищенность информации – это поддержание на заданном уровне тех параметров информации, которые характеризуют установленный статус ее хранения, обработки и использования. К таким параметрам можно отнести:

–*конфиденциальность*, свойство информации, указывающее на необходимость введения ограничений на круг субъектов, имеющих доступ к данной информации;

–*целостность*, способность вычислительной системы обеспечивать неизменность информации в условиях случайного и (или) преднамеренного искажения (разрушения);

–*доступность*, свойство системы, характеризующееся способностью обеспечивать своевременный беспрепятственный доступ субъектов к интересующей их информации.

Угрозы безопасности информации – события или действия, которые могут вызвать нарушение функционирования вычислительной системы, связанное с уничтожением или несанкционированным использованием обрабатываемой в ней информации.

Под ***доступом к информации*** понимается ознакомление с информацией, ее обработка, в частности, копирование, модификация или уничтожение информации.

Различают санкционированный и несанкционированный доступ к информации. *Санкционированный доступ* – это доступ к информации, не нарушающий установленные правила разграничения доступа. *Несанкционированный доступ* – доступ к информации, нарушающий установленные правила разграничения доступа с использованием штатных средств, предоставляемых вычислительной системой. Под штатными средствами понимается совокупность программного, микропрограммного и технического обеспечения средств вычислительной техники или автоматизированных систем. Несанкционированный доступ является одним из наиболее распространенных видов компьютерных нарушений.

Под ***атакой*** на вычислительную систему понимается действие, предпринимаемое злоумышленником, которое заключается в поиске и использовании той или иной уязвимости системы, т.е. атака – это реализация угрозы безопасности.

Политика безопасности - это совокупность норм, правил и практических рекомендаций, регламентирующих работу средств защиты ИС от заданного множества угроз безопасности.

Рассмотрим основные угрозы безопасности вычислительной системы.

1.2 Основные угрозы информационной безопасности

Известно большое количество разноплановых угроз безопасности информации различного происхождения, в литературных источниках предлагается несколько подходов к их классификации. При этом в качестве критериев деления множества угроз на классы используются виды порождаемых опасностей, степень злого умысла, источники проявления угроз и т.д. [5, 12, 14, 15, 21, 30].

Согласно одному из подходов, по цели воздействия можно выделить три основных типа угроз:

–угрозы нарушения конфиденциальности информации;

–угрозы нарушения целостности информации;

–угрозы нарушения работоспособности системы.

Угрозы нарушения конфиденциальности направлены на разглашение конфиденциальной или секретной информации, т.е. на изменение ее конфиденциального статуса. При реализации этих угроз информация становится известной лицам, которые не должны иметь к ней доступ. Угроза нарушения конфиденциальности имеет место всякий раз, когда получен несанкционированный доступ к некоторой закрытой информации, хранящейся в вычислительной системе или передаваемой от одной вычислительной системы к другой.

Угрозы нарушения целостности информации, хранящейся в вычислительной системе или передаваемой по каналам связи, направлены на ее изменение или искажение, приводящее к нарушению ее качества или полному уничтожению. Целостность информации может быть нарушена умышленно, а также в результате объективных воздействий со стороны среды, окружающей систему. Отметим, что эта угроза наиболее актуальна для систем передачи информации, в частности для систем, использующих web-интерфейс для взаимодействия с пользователем.

Угрозы нарушения работоспособности (так называемый «отказ в обслуживании», DoS-атака) направлены на создание таких ситуаций, когда определенные преднамеренные действия либо снижают работоспособность вычислительной системы, либо блокируют доступ к некоторым ее компонентам или ресурсам. Блокирование доступа может быть постоянным или временным. Данная угроза может быть реализована распределенно, при этом суммарное количество запросов к системе

может значительно превысить максимально возможное количество запросов, которые может обработать система. Такой тип атак носит название DDoS-атаки.

Функционирование вычислительной системы можно рассматривать в разрезе циркуляции в ней информационных потоков. В общем случае можно рассматривать поток информации от некоторого источника к некоторому адресату. Данная модель оказывается применима и в случае анализа взаимодействия пользователей и вычислительной системы с помощью какой-либо коммуникационной сети, например, Интернет (рисунок 1.1). В данном случае источником и приемником данных может являться как пользователь системы, так и сама система.

Можно выделить следующие типы воздействий на информационный поток: прерывание, перехват, модификация и фальсификация [21]. Отметим, что подобным воздействиям могут подвергаться как основной информационный поток, которым в данном случае является обмен по протоколу HTTP, так и вспомогательные информационные потоки, такие как запросы службы DNS.

Прерывание – воздействие, при котором ресурс системы становится недоступным или непригодным к использованию либо уничтожается. При этом нарушается доступность информации. Примером таких воздействий является разрушение каналов связи пользователей с системой, а также вывод из строя компонентов системы, отвечающих за взаимодействие пользователей с системой.

Перехват – получение несанкционированного доступа к ресурсу или каналу передачи информации, предполагает нарушение конфиденциальности информации. Примером таких нарушений может служить перехват трафика в вычислительной сети.

Модификация – воздействие, приводящее к изменению потока данных. При этом нарушается целостность информации. Отметим, что в общем случае нарушитель может изменять поток, проходящий через него в любых направлениях. Примерами подобных нарушений является модификация трафика сети или изменение маршрутов следования данных.

Фальсификация – попытки нарушителя создать несанкционированный поток данных, выдавая его за легальный. При таком воздействии нарушается аутентичность информации. Примером могут служить попытки злоумышленника выдать себя за легального

пользователя системы, в том числе и попытки передачи данных от имени легального пользователя системы.

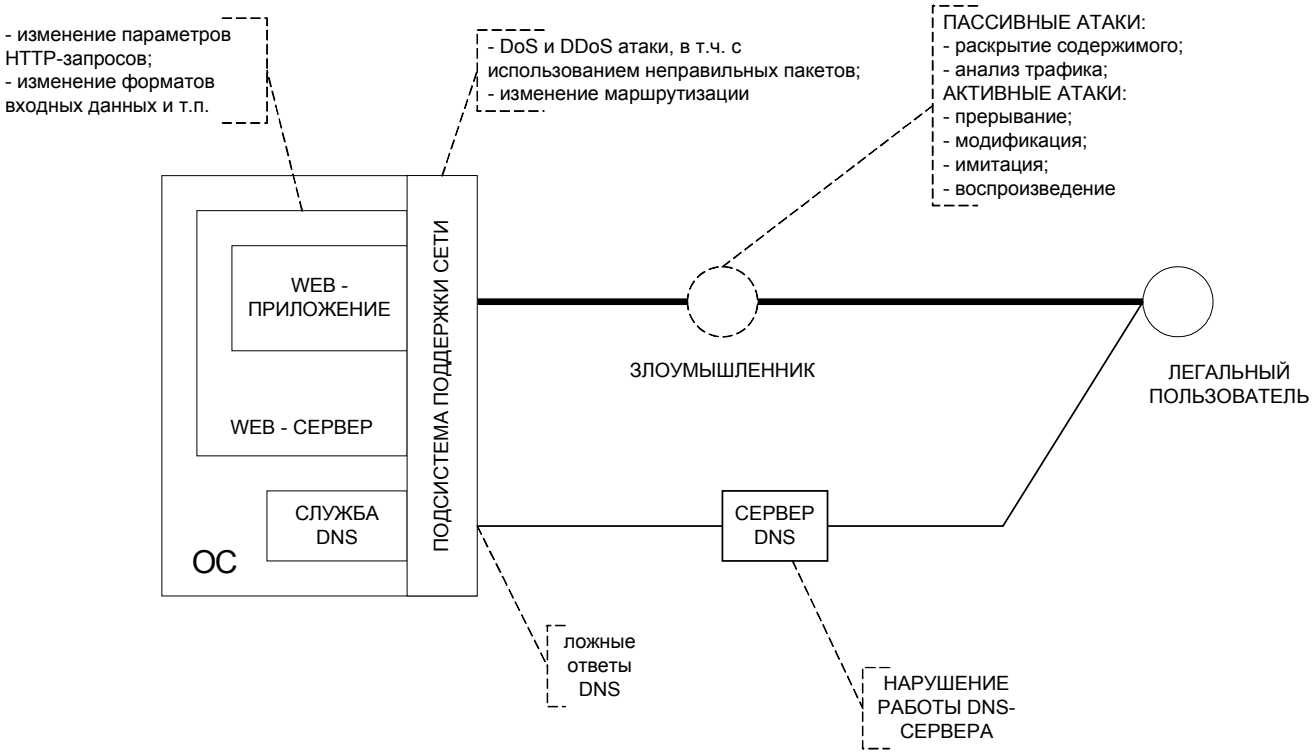

Рисунок 1.1 – Взаимодействие пользователей с вычислительной системой через глобальную сеть Интернет

Отметим, что все подобные атаки относятся к атакам типа «субъект в середине» (Man-in-Middle, MIM), то есть злоумышленник находится между взаимодействующими компонентами и получает доступ ко всем информационным потокам между ними.

Множество угроз безопасности можно разделить также на пассивные и активные угрозы безопасности [30].

Пассивные нарушения защиты информации имеют характер перехвата, или мониторинга, передаваемых данных. Целью нарушителя в этом случае является получение передаваемой информации. Пассивные нарушения можно условно разделить на раскрытие содержимого сообщений и анализ потока данных. Раскрытие сообщений предполагает ознакомление с передаваемой информацией в том виде, в каком с ней работают конечные пользователи, приложения и система.

При анализе потока данных злоумышленник не всегда может получить содержащуюся в сообщении информацию (например, если она зашифрована). Однако в этом случае остается возможность получить

некоторые характерные признаки сообщений. Например, злоумышленник может обнаружить и идентифицировать отправителя и используемые для отправки сообщений узлы, отследить частоту обмена сообщениями и их длину, попытаться установить причины обмена сообщениями.

Обнаружение пассивных нарушений защиты затруднено, поскольку они не предполагают каких-либо изменений данных, но в то же время возможно эффективное противодействие перехвату сообщений, например, с использованием средств аутентификации и криптографической защиты данных.

Активные нарушения связаны с изменением потока данных либо с созданием фальшивых потоков. Нарушения этого класса могут быть разделены на четыре группы: прерывание, воспроизведение, имитация, модификация.

Прерывание предполагает создание препятствия в нормальном функционировании средств связи отправителя и адресата или управлении ими. Например, объект может задерживать все сообщения, направленные определенному адресату. К данному классу относятся также воздействия, направленные на блокирование работы всей системы или сети передачи данных либо путем вывода ее из строя, либо путем преднамеренной загрузки системы, при которой происходит снижение производительности ее работы.

Воспроизведение представляет собой пассивный перехват блока данных и последующую повторную передачу этих данных с целью получения несанкционированных эффектов.

Имитация означает попытку одного объекта выдать себя за другой. Обычно имитация выполняется вместе с попыткой активного нарушения какого-либо другого типа – например, перехватив поток данных аутентификации, злоумышленник может воспроизвести реальную последовательность аутентификации, что позволит ему расширить свои полномочия, имитируя легальный объект, обладающий ими.

Модификация сообщений предполагает либо изменение части легального сообщения, либо его задержку, либо изменение порядка поступления сообщений с целью получения несанкционированного эффекта.

В отличие от пассивных нарушений активные достаточно легко обнаружить, и в то же время значительно труднее предотвратить. В случае активных нарушений вычислительная система должна как можно

быстрее обнаруживать нежелательные воздействия и восстанавливать свою нормальную работоспособность. Своевременное обнаружение нарушений может рассматриваться как часть системы предупреждения нарушений.

Как видно из приведенной на рисунке 1.1 схемы, объектами атак злоумышленника могут стать пользовательские компьютеры, каналы передачи данных, серверы DNS и серверный компьютер. В последнем случае возможны атаки, направленные на конкретное web-приложение, web-сервер в целом, СУБД (если она допускает подключения из внешней среды), а также другие сервисы (например, SMTP, FTP, службу DNS).

В общем, при взаимодействии пользователя и вычислительной системы через сеть Интернет возникают угрозы целостности, конфиденциальности, аутентификации и угрозы отказа в обслуживании (таблица 1.1) [39].

Таблица 1.1 – Угрозы нарушения защиты web-ресурсов

	Угрозы	**Последствия**
1	**2**	**3**
Целостность	–изменение пользовательских данных, –внедрение «троянских коней», –изменение информации в памяти, –изменение потока сообщений на пути их передачи	–потеря информации, –компрометация системы, –уязвимость в отношении угроз защиты остальных типов
Конфиден-циальность	–перехват данных в сети, –кража информации, хранящейся на сервере, –кража информации, хранящейся на компьютере-клиенте, –получение информации о конфигурации сети, –получение информации о клиенте, обращающемся к серверу	потеря информации, нарушение конфиденциальности информации

Продолжение таблицы 1.1

1	2	3
Отказ в обслуживании	–прекращение сеанса доступа пользователя, –перегрузка системы потоком фальшивых попыток доступа, –умышленное переполнение дискового пространства или оперативной памяти, –изоляция системы путем атак на DNS-сервер	–нарушение работы системы, –разрушение системы, –прекращение нормальной работы с пользователями
Аутентификация	–попытки нарушителя выдать себя за легального пользователя, –фальсификация данных	–неправильное представление пользователей, –доверие к искаженным данным

Угрозы безопасности можно также классифицировать по месту их возникновения: на компьютере-сервере (в частности, в web-приложении), на компьютере клиента (например, в web-браузере), в канале передачи данных между клиентом и сервером.

1.3 Основные атаки на вычислительные сети

Множество угроз информационной безопасности, возникающих при работе в вычислительной сети, можно условно разделить по уровням модели OSI, на которых реализуется та или иная угроза (рисунок 1.2). [14, 26, 30, 39, 41]

В работе рассматриваются угрозы безопасности и соответствующие методы защиты информации на уровнях выше канального. Это связано с тем, что при использовании общедоступной глобальной сети (которой и является Интернет) обеспечить физическую защиту каналов передачи данных практически невозможно.

Сетевой уровень модели OSI обеспечивает передачу данных между взаимодействующими узлами сети, отвечает за маршрутизацию, фрагментацию и дефрагментацию пакетов и т.п., независимо от

использующейся физической среды передачи данных и топологии сети. Рассмотрим атаки, возможные на этом уровне.

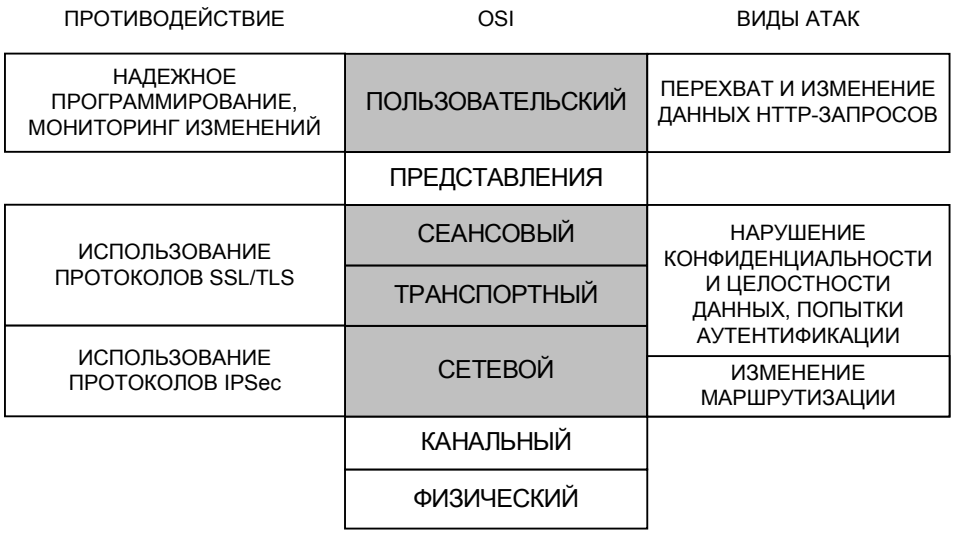

Рисунок 1.2 – Разделение угроз безопасности и средств защиты по уровням модели OSI

Анализ сетевого трафика является одним из наиболее распространенных типов атак. Проведение таких атак возможно в силу того, что многие сети, например, Ethernet, позволяют получить доступ к передающимся по ним данным любым их узлам. Кроме того, возможно несанкционированное подключение к сети. Примером такой атаки может служить перехват имени пользователя и пароля, передающихся по сети в незашифрованном виде. Таким образом, атака данного типа заключается в получении несанкционированного доступа к информации, которой обмениваются два сетевых абонента.

При использовании на сетевом уровне протокола IP возникает угроза того, что нарушитель может попытаться выдать себя за какой-либо доверенный объект сети, т.е. оказывается возможной атака, заключающаяся в передаче по каналам связи сообщений от имени произвольного субъекта или объекта сети. Существуют две разновидности данной атаки: атака при установленном виртуальном канале и атака без установленного виртуального канала. В случае установленного виртуального канала атака будет заключаться в присвоении прав доверенного субъекта взаимодействия, легально подключившегося к объекту системы, что позволит атакующему вести сеанс работы с объектом системы от имени доверенного субъекта.

Реализации данной атаки, как правило, основаны на том факте, что для аутентификации субъекта в протоколе IP задействован сетевой адрес (IP-адрес), который можно легко изменить. Для служебных сообщений часто используются передача одиночных сообщений, не требующих подтверждения, т.е. не требующих создания виртуального соединения. Атака без установленного виртуального соединения заключается в передаче служебных сообщений от имени сетевых управляющих устройств, например, маршрутизаторов. Посылка ложных управляющих сообщений может привести к серьезным нарушениям работы системы.

На данном уровне возможна также такая атака, как внедрение в распределенную вычислительную систему ложного объекта путем навязывания ложного маршрута.

Глобальные сети представляют собой совокупность сегментов сети, связанных между собой через сетевые узлы. При этом маршрутом называется последовательность узлов сети, по которой данные передаются от источника к приемнику. Каждый маршрутизатор имеет так называемую таблицу маршрутизации, в которой указываются оптимальные маршруты для каждого адресата. Таблицы маршрутизации существуют не только у маршрутизаторов, но и у любых компьютеров (хостов) в локальной сети. Для обеспечения эффективной и оптимальной маршрутизации в глобальных сетях используются специальные управляющие протоколы, позволяющие маршрутизаторам обмениваться информацией друг с другом (например, RIP – Routing Internet Protocol), уведомлять о новом маршруте (ICPM – Internet Control Message Protocol), удаленно управлять маршрутизаторами (SNMP – Simple Network Management Protocol). Все указанные выше протоколы позволяют удаленно изменить маршрутизацию в Internet и являются протоколами управления сетью. Таким образом, маршрутизация в глобальных сетях играет важную роль и, следовательно, может являться объектом атаки. Основная цель атаки, связанной с навязыванием ложного маршрута, состоит в том, чтобы изменить исходную маршрутизацию на объекте вычислительной системы таким образом, чтобы новый маршрут проходил через ложный объект – компьютер атакующего. Реализация данной атаки состоит в несанкционированном использовании протоколов управления сетью для изменения таблиц маршрутизации. Данная атака может осуществляться как в пределах одного сегмента, так и межсегментно. В случае успешной атаки злоумышленник получит полный контроль над

потоком информации, которой обмениваются два объекта распределенной вычислительной системы, и атака перейдет в стадию приема, анализа и передачи сообщений, получаемых от дезинформированных объектов системы.

Одной из возможных атак является внедрение в распределенную вычислительную систему ложного объекта путем использования недостатков алгоритмов удаленного поиска. В таких вычислительных системах часто оказывается, что объекты изначально не имеют достаточной информации для адресации сообщений. Обычно такой информацией являются аппаратные (например, адрес сетевого адаптера) и логические (IP-адрес) адреса объектов. Для получения такой информации используются различные алгоритмы удаленного поиска, заключающиеся в передаче по сети специальных поисковых запросов и ожидании на них ответов. После получения ответа на запрос субъект вычислительной системы обладает всеми необходимыми данными для адресации и, руководствуясь полученными данными об объекте, начинает адресоваться к нему. Примерами таких запросов в сети Internet могут служить ARP и DNS-запросы.

В случае использования механизмов удаленного поиска существует возможность на атакующем объекте перехватить посланный запрос и послать на него ложный ответ, где указать данные, использование которых приведет к адресации на атакующий ложный объект. В дальнейшем весь поток информации между субъектом и объектом взаимодействия будет проходить через ложный объект распределенной вычислительной системы.

Другой вариант внедрения ложного объекта использует недостатки алгоритма удаленного поиска и состоит в периодической передаче на атакуемый объект заранее подготовленного ложного ответа без приема поискового запроса. При этом атакующий может сам спровоцировать атакуемый объект на передачу поискового запроса, и тогда его ложный ответ будет немедленно иметь успех. Данная схема характерна для глобальных сетей, когда у атакующего из-за нахождения в другом сегменте отсутствует возможность перехвата поисковых запросов.

Отметим, что данная атака реализуется и на прикладном уровне модели OSI.

Получив контроль над проходящим потоком информации, ложный объект может применять различные методы воздействия на нее,

используя незащищенность протоколов верхнего уровня модели OSI (например, отсутствие средств аутентификации на уровне протокола TCP или отсутствие шифрования данных на уровне HTTP или FTP).

Кроме уже упоминавшихся возможностей раскрытия содержания информации, ее модификации и подмены, существует также возможность модификации передаваемого кода. При этом ложный объект, проводя семантический анализ проходящего через него кода, может выделять из потока данных исполняемый код. Для определения характера передаваемой по сети информации необходимо использовать особенности, свойственные реализации сетевого обмена в данной конкретной вычислительной системе или особенности, присущие исполняемым файлам в использующейся операционной системе. Отметим, что под исполняемым файлом понимаются как бинарные файлы операционной системы, так и различного вида сценарии, которые может выполнить операционная система или ее приложения.

Можно выделить два вида модификации кода, имеющих разные цели: внедрение разрушающих программных средств и изменение логики работы исполняемого файла.

В первом случае исполняемый файл модифицируется по вирусной технологии: одним из известных способов к файлу дописывается тело разрешающего программного средства, а также изменяется точка входа таким образом, чтобы она указывала на начало внедренного кода. Описанный способ отличается от обычного процесса инфицирования вирусом тем, что модификация исполняемого кода происходит при передаче его по сети. Такое воздействие возможно только при использовании ложного объекта, и может применяться, например, для создания сетевого червя.

Во втором случае происходит модификация исполняемого кода с целью изменения его работы. Данное воздействие требует предварительного исследования работы исполняемого файла, и в случае его проведения может преследовать различные цели. Например, возможно изменение процедуры аутентификации с целью получения несанкционированного доступа.

Одной из задач, возлагаемых на сетевую операционную систему, функционирующую на каждом из объектов распределенной вычислительной системы, является обеспечение надежного удаленного доступа для любого субъекта сети к любому объекту. Обычно для

предоставления удаленного доступа используются программы-серверы, запускаемые на объекте распределенной вычислительной системы. Задача сервера состоит в постоянном ожидании запросов от клиентов, и в случае получения таких запросов реагировать на них (например, передавать ответ, в котором либо разрешать, либо запрещать подключение) [14, 39].

Очевидно, что операционная система и программа-сервер могут допускать ограниченное число подключений, существующих в определенный момент времени. Это число зависит как от конфигурации самой системы, так и от пропускной способности каналов связи. Если в распределенной вычислительной системе не предусмотрено никаких средств аутентификации отправителя, т.е. она допускает передачу на атакуемый объект неограниченного числа анонимных запросов на подключение от других объектов, то в этом случае будет иметь успех атака, называемая отказом в обслуживании (DoS – Denied of Service). Результатом этой атаки является нарушение на атакованном объекте работоспособности соответствующей службы предоставления удаленного доступа или работоспособности самого объекта.

Второй разновидностью такой атаки является передача с одного адреса большого числа запросов на атакуемый объект. В этом случае, если не предусмотрено ограничения числа запросов с одного определенного адреса в единицу времени, возможно, переполнение очереди запросов одной из телекоммуникационных служб или отказ операционной системы вследствие нехватки системных ресурсов.

Еще одной разновидностью подобной атаки является передача на атакуемый объект специально сформированного некорректного запроса. В этом случае также возможно нарушение работы системы или телекоммуникационной службы.

На пользовательском уровне модели OSI возможно проведение атак, связанных с незащищенностью многих базовых протоколов взаимодействия, например, HTTP или FTP. Как правило, в этих протоколах не предусмотрено каких-либо средств обеспечения конфиденциальности передающейся информации, что позволяет осуществить не только ее перехват и ознакомление с ней, но также и модифицировать передающиеся данные. Кроме того, для протокола HTTP возможны различные манипуляции с полями заголовка, которые могут привести к нежелательным последствиям. Многие web-приложения

используют различные идентификаторы сеанса и маркеры Cookie, которые также могут быть перехвачены злоумышленником с целью выполнения каких-либо действий от имени другого пользователя системы. Злоумышленник может манипулировать значениями полей ввода, которые передаются web-приложению, что может также повлечь за собой негативные последствия.

1.4 Подходы к проектированию подсистемы защиты информации

Отметим, что создание подсистемы защиты информации, обеспечивающей противодействие всем возможным угрозам безопасности, зачастую является нецелесообразным. Это связано, например, с большими затратами на ее разработку, внедрение, поддержку и обеспечение ее функционирования. Кроме того, с увеличением числа угроз, которым противодействует подсистема информационной безопасности, растет и количество потребляемых ею ресурсов, что может само по себе негативно влиять на функционирование вычислительной системы. В связи с этим обычно выделяются наиболее вероятные или опасные для системы угрозы, защиту от которых и должна обеспечить подсистема информационной безопасности. [21, 30]

Подсистема защиты информации создается с учетом определенного набора норм, правил и практических рекомендаций, регламентирующих порядок обработки и хранения информации в вычислительной системе, называемого политикой безопасности. Политика безопасности, обеспечивающая достаточный уровень защищенности вычислительной системы, называется адекватной политикой безопасности [29].

Основной задачей при проектировании подсистемы защиты информации является выбор оптимальной политики безопасности, которая не только обеспечивает защиту от заданного множества угроз, но и не приводит к нежелательным последствиям.

Основным назначением вычислительной системы является переработка (сбор, хранение, обработка и выдача) информации, поэтому проблема обеспечения защиты информации является для нее центральной. Обеспечение безопасности вычислительной системы

предполагает организацию противодействия любому несанкционированному вторжению в процесс ее функционирования, а также попыткам модификации, хищения, выведения из строя или разрушения ее компонентов, т.е. защиту всех компонентов вычислительной системы - аппаратных средств, программного обеспечения, данных и персонала.

Существуют два подхода к проблеме обеспечения безопасности вычислительной системы: фрагментарный и комплексный.

Фрагментарный подход направлен на противодействие четко определенным угрозам в заданных условиях. В качестве примеров реализации такого подхода можно указать отдельные средства управления доступом, автономные средства шифрования, специализированные антивирусные программы и т.п.

Достоинством такого подхода является высокая избирательность к конкретной угрозе. Существенным недостатком данного подхода является отсутствие единой защищенной среды обработки информации. Фрагментарные меры защиты информации обеспечивают защиту конкретных объектов вычислительной системы только от конкретной угрозы. Даже небольшое видоизменение угрозы ведет к потере эффективности защиты.

Комплексный подход ориентирован на создание защищенной среды обработки информации в вычислительной системе, объединяющей в единый комплекс разнородные меры противодействия угрозам. Организация защищенной среды обработки информации позволяет гарантировать определенный уровень безопасности вычислительной системы, что является несомненным достоинством комплексного подхода. К недостаткам этого подхода относятся: ограничения на свободу действий пользователей, большая чувствительность к ошибкам установки и настройки средств защиты, сложность управления.

Объективная необходимость постановки проблемы комплексной защиты информации обусловлена системным характером влияния на ее безопасность большой совокупности различных обстоятельств, имеющих к тому же различную физическую природу и различные целевые посылки. Очевидно, что в этих условиях адекватным современным потребностям и условиям защиты информации может быть только комплексный подход к решению данных проблем.

Под комплексной защитой информации будем понимать целенаправленное применение в системах ее обработки различных средств, методов и мероприятий с целью поддержания заданного уровня защищенности информации по всей совокупности показателей и условий, являющихся существенно значимыми с точки зрения ее безопасности.

Система комплексной защиты информации предполагает:

–обеспечение физической целостности, т. е. заданной синтаксической структуры защищаемой информации;

–обеспечение логической целостности, т. е. семантических характеристик информации и установленных взаимосвязей между ее элементами;

–обеспечение доверия к информации в прагматическом плане, т. е. предупреждение ее несанкционированной модификации даже при сохранении синтаксических и семантических характеристик;

–предупреждение несанкционированного получения защищаемой информации лицами, не имеющими на это специальных полномочий, т. е. обеспечение установленного статуса ее конфиденциальности;

–предупреждение несанкционированного копирования информации, объявленной чьей-либо собственностью.

Комплексный подход к проблеме обеспечения безопасности основан на разработанной для конкретной вычислительной системы политике безопасности, которая регламентирует эффективную работу средств защиты системы.

Политика безопасности реализуется посредством административно-организационных мер, физических и программно-технических средств и определяет архитектуру системы защиты. Для конкретной организации политика безопасности должна носить индивидуальный характер и зависеть от конкретной технологии обработки информации и используемых программных и технических средств.

По способам осуществления все меры обеспечения безопасности можно подразделить на [23]:

–правовые;

–морально-этические;

–административные;

–физические;

–аппаратно-программные.

К правовым мерам защиты информации относятся действующие в стране законы, указы и другие нормативные документы, регламентирующие правила обращения с информацией ограниченного использования и ответственности за их нарушения. В РФ действуют следующие законы и нормативные документы РФ, касающиеся обеспечения защиты информации [27]:

–Закон РФ от 5 марта 1992 года N 2446-1 "О безопасности";

–Закон РФ от 23 сентября 1992 г. N 3523-1 "О правовой охране программ для электронных вычислительных машин и баз данных";

–Закон РФ от 21 июля 1993 года N 5485-1 "О Государственной тайне";

–Закон РФ от 20 февраля 1995 г. N 24-ФЗ "Об информации, информатизации и защите информации" (с комментариями);

–Закон РФ от 4 июля 1996 года N 85-ФЗ "Об участии в международном информационном обмене";

–Закон РФ от 10 января 2002 года N 1-ФЗ "Об электронной цифровой подписи";

–Закон РФ от 27 декабря 2002 года N 184-ФЗ "О техническом регулировании";

–Закон РФ от 7 июля 2003 года N 126-ФЗ "О связи";

–Закон РФ от 29 июля 2004 года N 98-ФЗ "О коммерческой тайне";

–Закон РФ от 8 августа 2001 года N 128-ФЗ "О лицензировании отдельных видов деятельности".

–Доктрина информационной безопасности РФ от 9 сентября 2000 г.;

–Указ Президента РФ от 31 декабря 1993г. N 2334 "О дополнительных гарантиях прав граждан на информацию";

–Указ Президента РФ от 20 января 1994г. N 170 "Об основах государственной политики в сфере информатизации";

–Указ Президента РФ от 3 апреля 1995 г. N 334 "О мерах по соблюдению законности в области разработки производства, реализации и эксплуатации шифровальных средств, а также предоставления услуг в области шифрования информации";

–Указ Президента РФ от 30 ноября 1995г. N 1203 "Об утверждении перечня сведений, отнесенных к государственной тайне";

–Указ Президента РФ от 6 марта 1997 г. N 188 "Об утверждении перечня сведений конфиденциального характера";

–Указ Президента РФ от 30 мая 1997 г. N 226-рп "О перечне должностных лиц органов государственной власти, наделяемых полномочиями по отнесению сведений к государственной тайне";

–Указ Президента РФ от 11 августа 2003 г. N 960 "Вопросы Федеральной Службы Безопасности Российской Федерации";

–Указ Президента РФ от 12 мая 2004 года N 611 "О мерах по обеспечению информационной безопасности Российской Федерации в сфере международного информационного обмена";

–Постановление Правительства РФ от 15 апреля 1995 года N 333 "О лицензировании деятельности предприятий, учреждений и организаций по проведению работ, связанных с использованием сведений, составляющих государственную тайну, созданием средств защиты информации, а также с осуществлением мероприятий и (или) оказанием услуг по защите государственной тайны"

–Постановление Правительства РФ от 26 июня 1995 г. N 608 "О сертификации средств защиты информации";

–Постановление Правительства РФ от 04 сентября 1995г. N 870 "Об утверждении правил отнесения сведений, составляющих государственную тайну, к различным степеням секретности";

–Постановление Правительства РФ от 30 апреля 2002 г. N 290 "О лицензировании деятельности по технической защите конфиденциальной информации";

–Постановление Правительства РФ от 27 мая 2002 г. N 348 «Об утверждении положения о лицензировании деятельности по разработке и (или) производству средств защиты конфиденциальной информации».

К морально-этическим мерам противодействия относятся всевозможные нормы поведения, эти нормы не являются обязательными, но их несоблюдение ведет к падению престижа личности или организации.

Административные меры защиты информации относятся к мерам организационного характера. Они регламентируют:

–процесс функционирования вычислительной системы;

–использование ресурсов;

–деятельность персонала;

–порядок взаимодействия пользователей с системой.

Административные меры включают:

–разработку правил обработки информации в вычислительной системе;

–организацию надежного пропускного режима;

–организацию учета, хранения, использования и уничтожения документов и носителей с конфиденциальной информацией и т.п.

Несмотря на то, что вышеперечисленные меры безопасности не будут рассмотрены в дальнейшем в пособии, не стоит забывать о важности их проектирования и применения. Важно отметить, что пока не будут реализованы действенные меры административной защиты, прочие меры будут неэффективны.

К физическим мерам защиты относятся разного рода механические, электромеханические и электронно-механические устройства, специально предназначенные для создания физических препятствий на возможных путях проникновения и доступа потенциальных нарушителей к компонентам системы и защищаемой информации.

К аппаратно-программным мерам защиты информации относятся электронные устройства и специальные программы, которые реализуют самостоятельно или в комплексе с другими средствами следующие способы защиты:

–идентификацию и аутентификацию субъектов доступа;

–разграничение доступа к ресурсам вычислительной системы;

–контроль целостности данных;

–регистрацию и анализ событий, происходящих в вычислительной системе;

–резервирование ресурсов и компонентов вычислительной системы.

2 Криптографические системы

2.1 Развитие криптографических систем

Криптография представляет собой совокупность методов преобразования данных, направленных на сокрытие смысла сообщения с помощью шифрования и открытие его расшифровыванием, которые выполняются по специальным криптографическим алгоритмам с помощью ключей отправителя и получателя.

Такие преобразования позволяют решить две главные проблемы защиты данных: проблему конфиденциальности (путем лишения злоумышленника возможности получения смысла информации, передаваемой по каналам связи) и проблему целостности (путем лишения злоумышленника возможности изменить сообщение или ввести ложное сообщение в канал связи) без знания ключевой информации.

В истории развития криптографии можно выделить три основных периода (рисунок 2.1). Эти периоды выделены достаточно условно и говорят лишь о времени появления шифров, многие из них используются и в настоящее время. [1, 16]

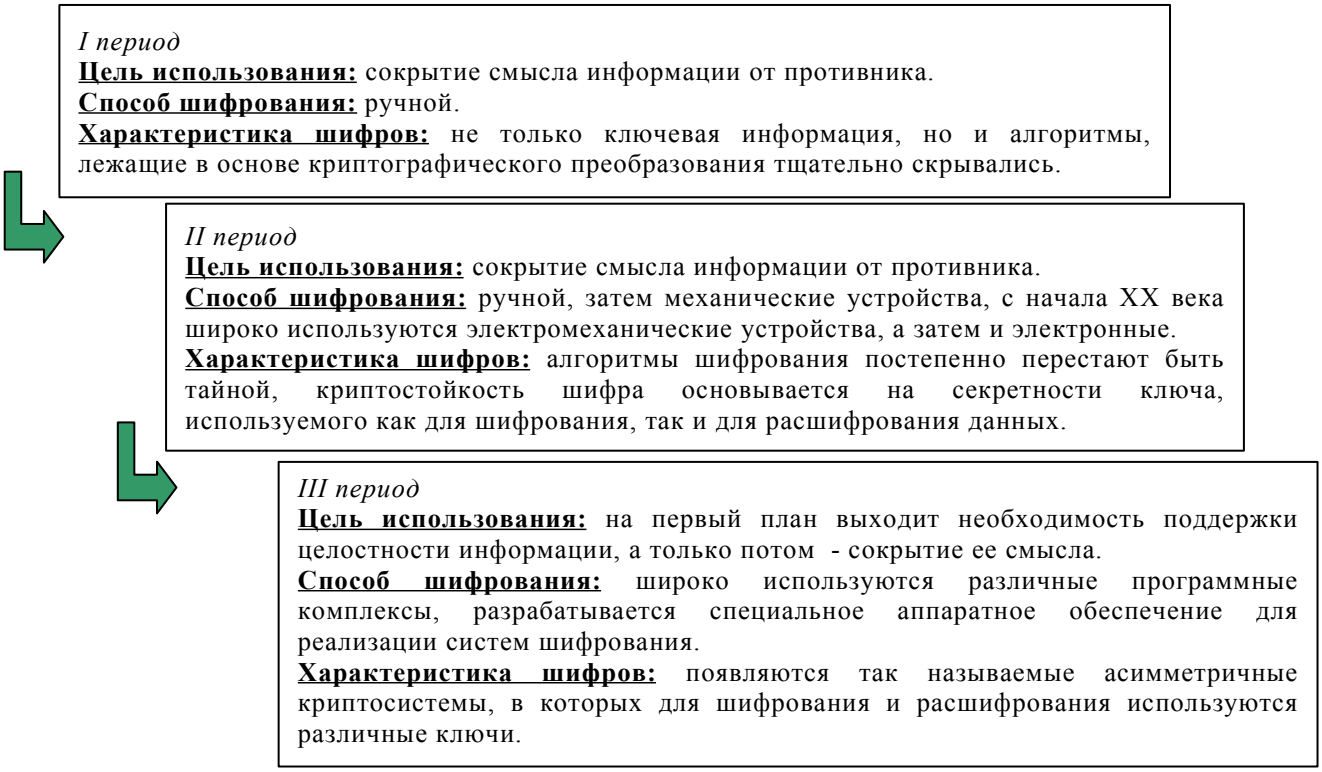

Рисунок 2.1

I период – начальный, имевший дело лишь с ручными шифрами. Точную дату появления первой системы шифрования назвать достаточно трудно. Один из первых шифров – скитала – использовался в Спарте (V в. до н.э.). Закончился данный период приблизительно в конце тридцатых годов XX века в связи с появлением механических, а затем и электронных устройств шифрования. Следует отметить, что отдельные элементы, разработанные за это время, – так называемые традиционные системы шифрования – широко используются и в современных алгоритмах шифрования.

II период отмечен созданием и широким внедрением в практику сначала механических, потом электромеханических и, наконец, электронных устройств шифрования, созданием сетей засекреченной связи. Его началом можно считать применение телеграфных шифровальных машин, использующих длинный одноразовый ключ. Однако к середине семидесятых годов было достигнуто положение, когда повышение стойкости шифров отошло на второй план. С развитием разветвленных коммерческих сетей связи, электронной почты и глобальных информационных систем самыми главными стали проблемы распределения секретных ключей и подтверждения авторства. К ним теперь привлечено внимание широкого круга криптологов.

III период. Началом третьего периода развития криптологии обычно считают 1976 год, когда американские математики Диффи и Хеллман предложили принципиально новый вид организации засекреченной связи без предварительного снабжения абонентов секретными ключами, так называемое шифрование с открытым ключом (асимметричное шифрование). В результате стали появляться криптографические системы, основанные на подходе, сформулированном еще в сороковых годах Шенноном. В основе этого подхода лежит идея построения шифра таким способом, чтобы его раскрытие было эквивалентно решению математической задачи, требующей выполнения объемов вычислений, превосходящих возможности современных ЭВМ. Новый период развития криптографии характеризуется появлением полностью автоматизированных систем шифрованной связи, в которых каждый пользователь имеет свой индивидуальный пароль для подтверждения подлинности и предъявляет при входе в систему, а весь остальной процесс проведения секретной связи происходит автоматически.

В общем виде процедуру шифрования данных можно представить следующим образом (рисунок 2.2). Отправитель генерирует открытый текст исходного сообщения M, которое должно быть передано законному получателю по незащищенному каналу, доступ к которому может получить злоумышленник с целью перехвата и раскрытия смысла передаваемого сообщения. Для того, чтобы злоумышленник не смог узнать содержание сообщения, отправитель шифрует его с помощью обратимого преобразования E и ключа K, тем самым получая шифртекст сообщения (криптограмму) $C = E_к(M)$, который затем отправляет получателю.

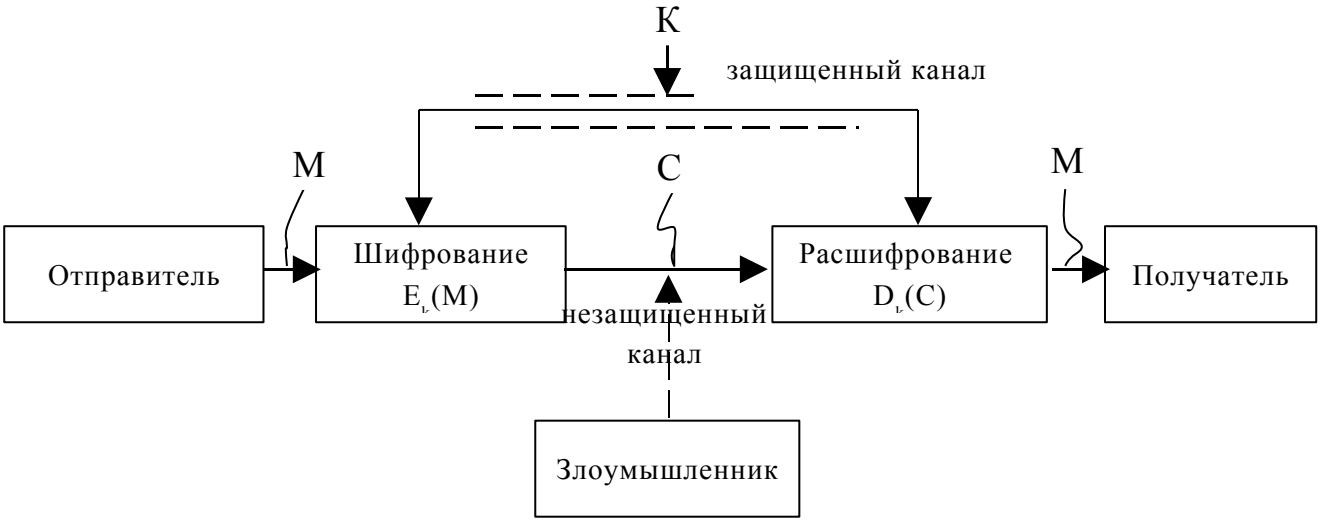

Рисунок 2.2

Получатель, приняв шифртекст C, расшифровывает его с помощью обратного преобразования $D=E_к^{-1}$ и получает исходное сообщение M.

$$D_k(M)=E_k^{-1}(E_k(M))=M$$

Преобразование $E_к$ выбирается из семейства криптографических преобразований, называемых криптоалгоритмами. Параметр, с помощью которого выбирается отдельное используемое преобразование, называется криптографическим ключом.

Преобразование шифрования может быть симметричным или асимметричным относительно преобразования расшифрования. Это важное свойство функции преобразования определяет два класса криптосистем:

- симметричные (одноключевые) криптосистемы;
- асимметричные (двухключевые) криптосистемы (или криптосистемы с открытым ключом).

Основной характеристикой шифра является криптостойкость, которая определяет его стойкость к раскрытию методами криптоанализа. Обычно эта характеристика определяется интервалом времени, необходимым для раскрытия шифра.

2.2 Симметричные криптографические системы

2.2.1 Традиционные криптографические симметричные системы

Характерной особенность симметричных систем шифрования является то, что для шифрования и расшифрования сообщений применяется один и тот же ключ, который вследствие этого является секретным и должен передаваться получателю по секретным каналам связи.

Симметричные алгоритмы шифрования можно представить следующим образом:

- шифры перестановок;
- шифры замены;
- шифры гаммирования;
- шифры, основанные на аналитических преобразованиях шифруемых данных.

Шифрование перестановкой заключается в том, что символы шифруемого текста переставляются по определенному правилу в пределах некоторого блока текста. При достаточной длине блока, в пределах которого осуществляется перестановка, и сложном неповторяющемся порядке перестановки можно достигнуть приемлемой стойкости шифра для практических приложений, не требующих выской степени защищенности данных.

Шифрование заменой (подстановкой) заключается в том, что символы шифруемого текста заменяются символами того же или другого алфавита в соответствии с заранее обусловленной схемой замены.

Шифрование гаммированием заключается в том, что символы шифруемого текста складываются с символами некоторой случайной последовательности, именуемой гаммой шифра (например, сложение по модулю 2).

Шифрование аналитическим преобразованием заключается в том, что шифруемый текст преобразуется по некоторому аналитическому правилу (формуле).

2.2.2 Шифры перестановки

Шифр перестановки «скитала» использовался в Спарте (V веке до н.э.). Свое название шифр получил от устройства, использовавшегося для шифрования/расшифрования данных – скитала (первого простейшего криптографического устройства). Процедура шифрования заключалась в следующем: полоска пергамента наматывалась спиралью на палочку (скиталу), вдоль которой выписывался текст сообщения. После снятия полоски буквы на ней расположатся хаотично. Для прочтения шифровки нужно не только знать систему засекречивания, но и обладать ключом в виде палочки, принятого диаметра. Зная тип шифра, но, не имея ключа, расшифровать сообщение было сложно.

Аристотелю принадлежит способ расшифрования этого шифра: необходимо изготовить конус и, начиная с основания, обертывать его лентой с зашифрованным сообщением, постепенно сдвигая ее к вершине. Когда начнут просматриваться куски сообщения, можно будет определить диаметр скиталы.

Шифрующие таблицы. В ручных шифрах часто используются таблицы, которые дают простые шифрующие процедуры на основе перестановки букв в сообщении. Ключом в таких шифрах служат размер таблицы, фраза, задающая перестановку, маршрут перестановки или специальная особенность таблиц.

Простая перестановка без ключа - один из самых простых методов шифрования, смысл которого заключается в последовательном вписывании букв сообщения в ячейки таблицы, соответственно количество ячеек таблицы не должно быть меньше длины сообщения. Рассмотрим эту процедуру на примере шифрования сообщения ПРИХОДИТЕ_ЗАВТРА. Длина сообщения – 16 символов, в качестве размерности таблицы выбираем 4х4.

П	О	Е	В
Р	Д	_	А
И	И	З	Р
Х	Т	А	А

После того, как открытый текст записан колонками, для образования шифртекста он считывается по строкам: ПОЕВРД_АИИЗРХТАА. Для использования этого шифра отправителю и получателю нужно договориться об общем ключе в виде размера таблицы и порядке считывания символов.

Разновидностью рассмотренного выше метода является метод, называемый одиночной перестановкой по ключу. Он отличается лишь тем, что колонки таблицы переставляются по ключевому слову, фразе или набору чисел длиной в строку таблицы. Использовав в виде ключа слово РЫБА, получим такой результат:[1].

до перестановки

Р	**Ы**	**Б**	**А**
3	*4*	*2*	*1*
П	О	Е	В
Р	Д	_	А
И	И	З	Р
Х	Т	А	А

после перестановки

А	**Б**	**Р**	**Ы**
1	*2*	*3*	*4*
В	Е	П	О
А	_	Р	Д
Р	З	И	И
А	А	Х	Т

Получается шифртекст: ВЕПОА_РДРЗИИААХТ.

С целью увеличения криптостойкости можно повторно шифровать сообщение, которое уже было зашифровано. Этот способ известен под названием двойная перестановка. Для этого размер второй таблицы подбирают так, чтобы длины ее строк и столбцов были другие, чем в

[1] цифры в таблице указывают на алфавитный порядок букв

первой таблице. Лучше всего, если они будут взаимно простыми. Кроме того, в первой таблице можно переставлять столбцы, а во второй строки. Наконец, можно заполнять таблицу зигзагом, змейкой, по спирали или каким-то другим способом. Такие способы заполнения таблицы если и не усиливают стойкость шифра, то делают процесс шифрования гораздо более занимательным. [1]

Кроме одиночных перестановок использовались еще двойные перестановки столбцов и строк таблицы с сообщением. При этом перестановки определялись отдельно для столбцов и отдельно для строк. В таблицу вписывался текст и переставлялись столбцы, а потом строки.

Магические квадраты – квадратные таблицы со вписанными в их клетки последовательными натуральными числами начиная с 1. Сумма ячеек в каждом столбце, каждой строке и каждой диагонали представляет собой одно и тоже число. Подобные квадраты широко применялись для вписывания шифруемого текста по приведенной в них нумерации. Если потом выписать содержимое таблицы по строкам, то получалась шифровка перестановкой букв. Считалось, что созданные с их помощью шифровки охраняет не только ключ, но и магическая сила. Ниже приведен пример магического квадрата и его заполнение сообщением ПРИЛЕТАЮ ВОСЬМОГО:

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

О	И	Р	М
Е	О	С	Ю
В	Т	А	Ь
Л	Г	О	П

Шифртекст, получаемый при считывании содержимого правой таблицы по строкам: ОИРМЕОСЮВТАЬЛГОП

2.2.3 Шифры простой замены

При шифровании заменой (подстановкой) символы шифруемого текста заменяются символами того же или другого алфавита по заранее

установленному правилу замены. Часто шифры простой замены называют шифрами одноалфавитной подстановки. [30]

Шифр Цезаря - один из первых шифров замены. Суть шифрования заключалась в следующем: каждая буква шифровки заменялась четвертой по счету от нее в алфавите: A-B-C-D, или D вместо A. Послание сенату VENI VIDI VICI, то есть ПРИШЕЛ УВИДЕЛ ПОБЕДИЛ, сделанное Цезарем после однодневной войны с понтийским царем Фарнаком, выглядело бы шифровкой YHQL YLGL YLFL.

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C

Процесс шифрования простой заменой может быть описан формулой:

$$y_i=(x_i+k) \bmod m, \qquad (2.1)$$

где
i=1..n, n – количество букв в сообщении;
0 <=k<m, m – количество букв в алфавите сообщения;
mod - операция взятия остатка от деления.

Процесс расшифрования:

$$x_i=(y_i - k) \bmod m$$

Например, зашифруем сообщение ЮСТАС АЛЕКСУ, используя k=2.

Русский алфавит:

_ (пробел)	0	Г	4	З	8
А	1	Д	5	И	9
Б	2	Е	6	Й	10
В	3	Ж	7	К	11

Л	12	Т	19	Щ	26
М	13	У	20	Ъ	27
Н	14	Ф	21	Ы	28
О	15	Х	22	Ь	29
П	16	Ц	23	Э	30
Р	17	Ч	24	Ю	31
С	18	Ш	25	Я	32

Проставим соответствие:

Ю	31	А	1
С	18	Л	12
Т	19	Е	6
А	1	К	11
С	18	С	18
_	0	У	20

Подставляя данные в формулу (2.1), получим шифртекст:

31 + 2 = 33 mod 33 = 0	_
18 + 2 = 20 mod 33 = 20	У
19 + 2 = 21 mod 33 = 21	Ф
1 + 2 = 3 mod 33 = 3	В
18 + 2 = 20 mod 33 = 20	У
0 + 2 = 2 mod 33 = 2	Б
1 + 2 = 3 mod 33 = 3	В
12 + 2 = 14 mod 33 = 14	Н
6 + 2 = 8 mod 33 = 8	З
11 + 2 = 13 mod 33 = 13	М
18 + 2 = 20 mod 33 = 20	У
20 + 2 = 22 mod 33 = 22	Х

Шифртекст: _УФВУБВНЗМУХ

Основной недостаток данного шифра заключается в том, что сохраняется естественная статистика исходного языка. Известно, например, что в русском языке буква "Е" встречается чаще, чем буква "З". На основании анализа слов языка (различных текстов, документов) можно построить таблицы частот встречаемости букв алфавита в словах.

Построенная зависимость будет распространяться и на символы шифртекста, что дает некоторую информацию злоумышленнику об исходном тексте. Например, данный подход был применен в знаменитых "Пляшущих человечках" Артура Конан Дойля, а также в "Золотом жуке" Эдгара По.

2.2.4 Шифры сложной замены

Шифры сложной замены называют многоалфавитными (в отличие от одноалфавитных - шифров простой замены), так как для шифрования каждого символа исходного сообщения применяют свой шифр замены.

Шифры многоалфавитной замены обеспечивают маскирование естественной статистики исходного языка, поскольку конкретный символ исходного текста может быть преобразован в разные символы шифртекста.

Многоалфавитные шифры замены предложил и ввел в практику Леон Батист Альберти, который также был известным теоретиком искусства и архитектором. Его книга "Трактат о шифре", написанная в 1566 г, представляла собой первый в Европе научный труд по криптологии. [16]

Шифр Гронсфельда. Представляет собой модификацию шифра Цезаря числовым ключом. Для этого под буквами исходного сообщения записывают цифры числового ключа. Если ключ короче сообщения, то его запись циклически повторяют.

Шифртекст получают по принципу, используемому в шифре Цезаря, но отсчитывают теперь не третью букву по алфавиту, а ту, которая сдвинута на соответствующую цифру ключа. Так, применяя в качестве ключа последовательность цифр 253, получаем шифровку:

Сообщение	Д	Е	Л	О	_	З	А	К	Р	Ы	Т	О
Ключ	2	5	3	2	5	3	2	5	3	2	5	3
Шифртекст	Ж	К	О	Р	Д	К	В	П	У	Э	Ч	С

Чтобы зашифровать первую букву сообщения Д, используя первую цифру ключа 2, отсчитывается вторая по порядку от Д в алфавите буква Д-Е-Ж и получается буква шифровки Ж.

Система шифрования Вижинера. Является наиболее известной из многоалфавитных систем (по имени французского дипломата XVI века Блеза Вижинера). В основе данного шифра лежит специальная таблица шифрования, получившая название таблицы Вижинера.

В первой строке этой таблицы располагается исходный алфавит (дополненный пробелом). В первом столбце также выписываются буквы алфавита. Начиная со второй строки, для каждого символа первого столбца образуется свой алфавит путем циклического сдвига вправо алфавита предыдущей строки на один символ.

При шифровании сообщения его выписывают в строку, а под ним ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Шифровку получают, находя символ в столбце таблицы по букве сообщения и строке, соответствующей букве ключа.

В качестве примера рассмотрим шифрование сообщения ОТВЕТ_ПОЛУЧЕН, используя ключ ФИАЛКА.

сообщение	О	Т	В	Е	Т	_	П	О	Л	У	Ч	Е	Н
ключ	Ф	И	А	Л	К	А	Ф	И	А	Л	К	А	Ф
шифртекст	Ы	Л	В	Ы	И	_	Ь	Ж	Л	И	Н	Е	Ъ

Таким образом, можно сказать, что на каждом шаге система шифрования Вижинера представляет собой простую (одноалфавитную) замену. Отличие заключается в том, что ключ меняется от буквы к букве.

Шифры многоалфавитной замены могут быть описаны аналитически следующим образом. Открытый текст $x=(x_0, \ldots, x_{n-1})$ преобразуется в шифртекст $y=(y_0, \ldots, y_{n-1})$ с помощью ключа $k=(k_0, \ldots, k_t)$ согласно следующему правилу:

$$y_i = (x_i + k_j) \bmod m,$$

где

$i=0\ldots n-1$;

$j= i \bmod t$;

m - количество символов в алфавите.

Процесс расшифрования можно представить в виде:

$$x_i = (y_i - k_j) \bmod m$$

Например, зашифруем сообщение ПРИХОДИТЕ ЗАВТРА с ключом КОТ. Согласно таблице 2.1, представим исходное сообщение в числовом виде:

П	16	Е	6
Р	17	_	0
И	10	З	8
Х	22	А	1
О	15	В	3
Д	5	Т	19
И	10	Р	17
Т	19	А	1

Представим ключ также в цифровой форме:

К	11
О	15
Т	19

Получим шифртекст согласно вышеуказанным правилам:

16 + 11 = 27 mod 33 = 27	Ъ
17 + 15 = 32 mod 33 = 32	Я
10 + 19 = 29 mod 33 = 29	Ь
22 + 11 = 33 mod 33 = 0	_
15 + 15 = 30 mod 33 = 30	Э
5 + 19 = 24 mod 33 = 24	Ч
10 + 11 = 21 mod 33= 21	Ф
19 + 15 = 34 mod 33 = 1	А
6 + 19 = 25 mod 33 = 25	Ш
0 + 11 = 11 mod 33 = 11	К
8 + 15 = 23 mod 33 = 23	Ц
1 + 19= 20 mod 33 = 20	У
3 + 11= 14 mod 33 = 14	Н
19 + 15 = 34 mod 33 = 1	А
17 + 19 = 36 mod 33 = 3	В
1 + 11 = 12 mod 33 = 12	Л

Шифртекст: ЪЯЬ_ЭЧФАШКЦУНАВЛ

Таблица 2.1

	А	**Б**	**В**	**Г**	**Д**	**Е**	**Ж**	**З**	**И**	**Й**	**К**	**Л**	**М**	**Н**	**О**	**П**	**Р**	**С**	**Т**	**У**	**Ф**	**Х**	**Ц**	**Ч**	**Ш**	**Щ**	**Ъ**	**Ы**	**Ь**	**Э**	**Ю**	**Я**	**_**
А	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	
Б	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
В	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	
Г	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	
Д	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	
Е	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	
Ж	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	
З	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	
И	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	
Й	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	
К	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	
Л	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	
М	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	
Н	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	
О	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	
П	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	
Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	
С	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	
Т	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	
У	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	
Ф	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	
Х	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	
Ц	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	
Ч	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	Й	
Ш	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	И	
Щ	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	З	
Ъ	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	Е	Ж	
Ы	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я		А	Б	В	Г	Д	Е	
Ь	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	Д	
Э	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	Г	
Ю	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	В	
Я	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	Б	
_	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	_	А	

Таким образом, символ "Р", встречающийся дважды в исходном тексте, в шифртексте встречается как "Я" и как "В" (это замечание касается и других встречающихся несколько раз в исходном тексте символов). Может возникнуть ситуация, когда различные в исходном тексте символы будут зашифрованы одинаково.

2.2.5 Гаммирование

Под гаммированием понимают процесс наложения по определенному закону гаммы шифра на исходные данные. Гамма шифра – это псевдослучайная последовательность, выработанная по определенному алгоритму.

Процесс шифрования заключается в генерации гаммы шифра и наложения полученной гаммы на исходный открытый текст обратимым образом (обычно используется операция сложения по модулю 2).

$$c_i = m_i \oplus \gamma_i$$

Процесс расшифрования заключается в повторной генерации гаммы шифра и наложению этой гаммы на зашифрованные данные.

$$m_i = c_i \oplus \gamma_i$$

Получаемый таким методом шифртекст достаточно труден для раскрытия, поскольку его ключ является переменным. Если период гаммы превышает длину всего шифруемого текста и злоумышленнику неизвестна никакая часть исходного текста, то такой шифр можно раскрыть только прямым перебором всех вариантов ключа. В этом случае криптостойкость шифра будет определяться длиной ключа.

2.3 Современные симметричные криптографические системы

2.3.1 DES

Стандарт шифрования данных DES (Data Encryption Standard) был опубликован Национальным бюро стандартов США в 1977 году, а в 1980 году принят в качестве стандарта шифрования данных для защиты от несанкционированного доступа к важной, но несекретной информации в государственных и коммерческих организациях США. [1, 2, 30]

DES обладает рядом достоинств: высокой скоростью аппаратной и программной реализации, высокой криптостойкостью.

В общем виде функционирование алгоритма DES можно описать следующей схемой (рисунок 2.3).

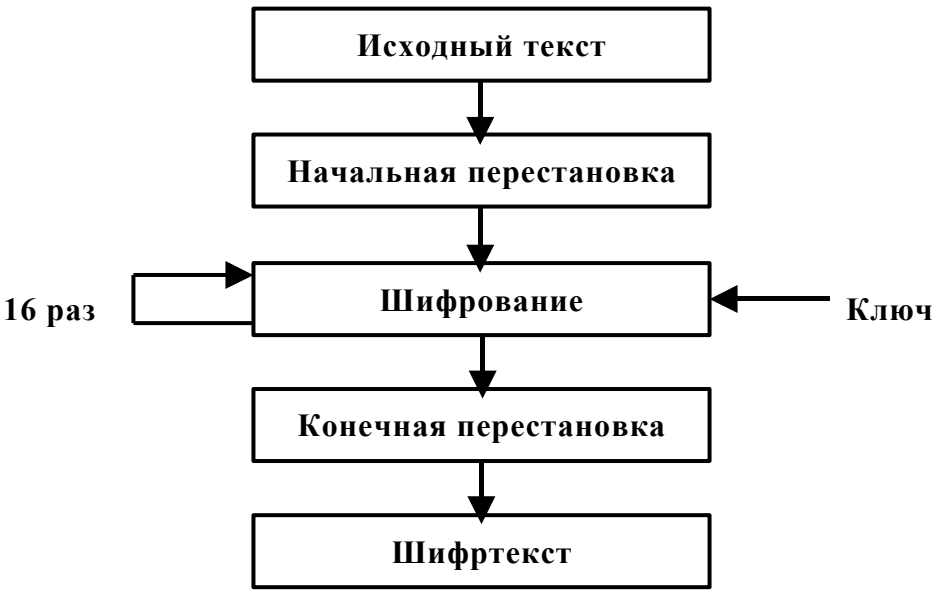

Рисунок 2.3

Более подробная схема алгоритма представлена на рисунке 2.4.

В алгоритме используются следующие обозначения:

L_i и R_i – левый и правый 32-битовые блоки исходного 64 битового сообщения.

$\oplus$ – операция побитового сложения двоичных блоков по модулю 2.

K_i – 48-битовый ключ.

f – функция шифрования.

IP – начальная перестановка для 64 битового блока сообщения.

Исходный текст

Алгоритм DES оперирует 64-битовыми двоичными блоками, на которые разбивается исходное сообщение. Размерность ключа, используемого для шифрования данных, равна 56 битам.

Начальная перестановка

64 битовый блок текста предварительно переставляется:

$$T_0 = IP(M)$$

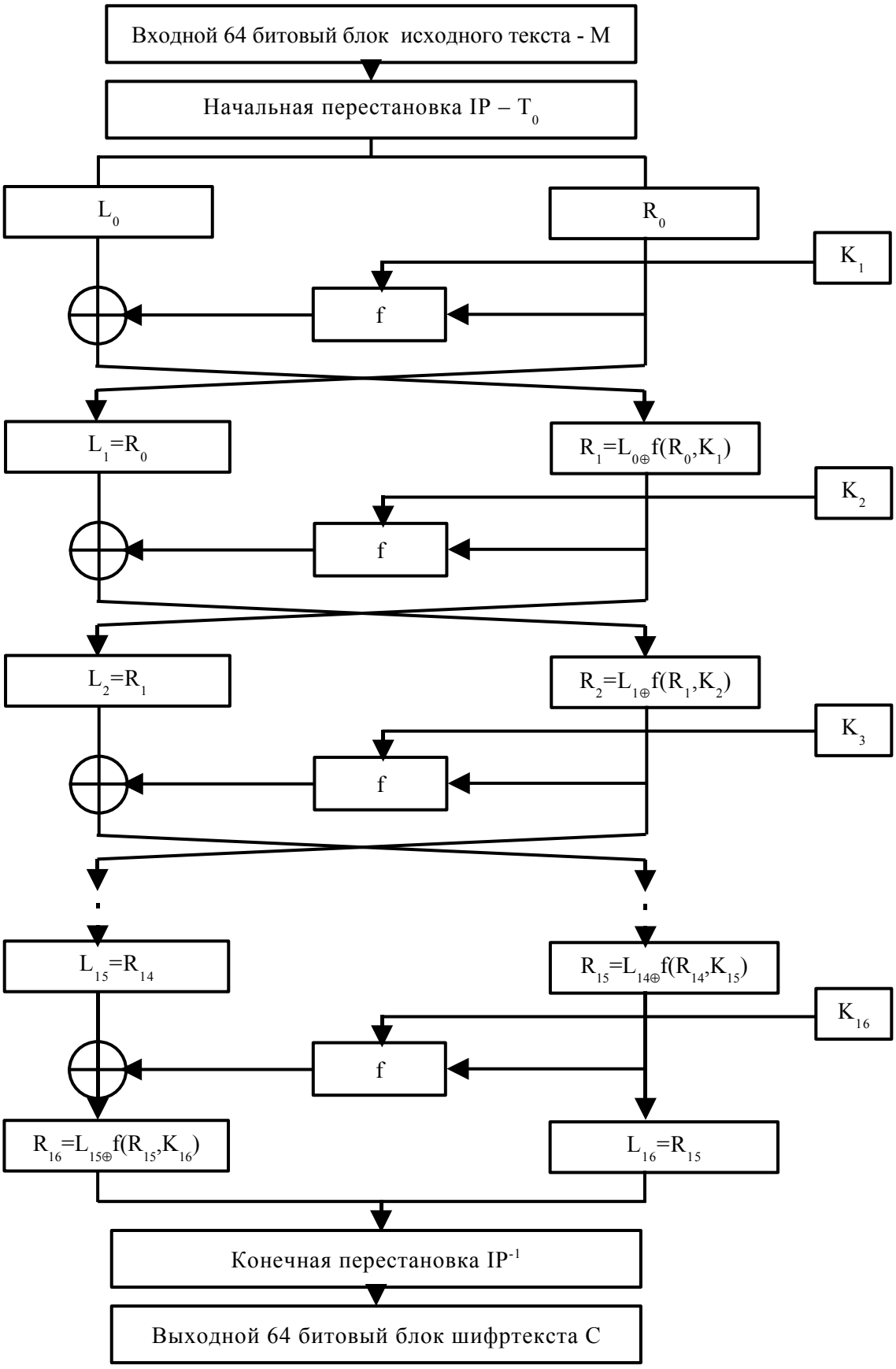
Входной 64 битовый блок исходного текста - M
Начальная перестановка IP – T_0
L_0
R_0
K_1
f
$L_1=R_0$
$R_1=L_{0\oplus}f(R_0,K_1)$
K_2
f
$L_2=R_1$
$R_2=L_{1\oplus}f(R_1,K_2)$
K_3
f
$L_{15}=R_{14}$
$R_{15}=L_{14\oplus}f(R_{14},K_{15})$
K_{16}
f
$R_{16}=L_{15\oplus}f(R_{15},K_{16})$
$L_{16}=R_{15}$
Конечная перестановка IP^{-1}
Выходной 64 битовый блок шифртекста C

Рисунок 2.4

Алгоритмы перестановки являются стандартными в том смысле, что они заданы в самом стандарте DES. Значения подобраны таким образом, чтобы затруднить процесс вскрытия, и должны использоваться во всех реализациях алгоритма DES в неизменном виде. Значения таблиц начальной и конечной перестановок приведены в таблицах 2.2 и 2.3.

Таблица 2.2 – Матрица начальной перестановки IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Таблица 2.3 – Матрица обратной перестановки IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

Шифрование

Полученный после перестановки блок сообщения разделяется на две части:

L_0 – 32 старших бита сообщения;

R_0 – 32 младших бита.

$$T_0 = L_0R_0$$

После этого выполняются 16 циклов преобразования по следующим формулам.

Пусть

$$T_{i-1} = L_{i-1}R_{i-1}$$

для i=1..16.

Тогда на i-ом шаге

$$T_i = L_i R_i$$

$$L_i = R_{i-1}$$

$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

Для вычисления функции f используется следующая схема (рисунок 2.5).

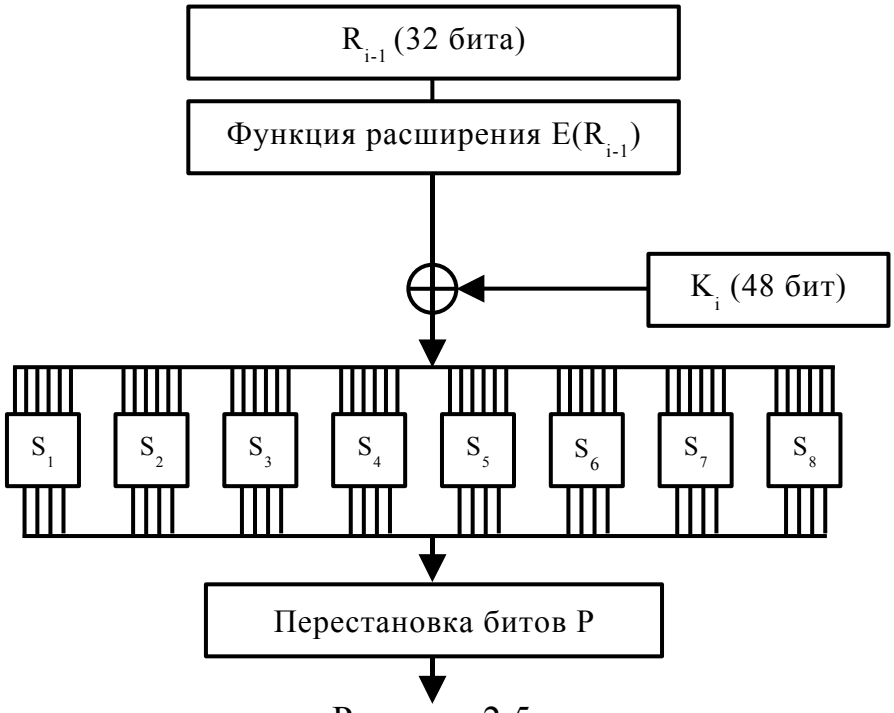

Рисунок 2.5

Как видно из рисунка 2.5, при вычислении функции f используется функция E, задача которой расширить исходное 32 битовое сообщение до 48 бит. Расширение происходит за счет дублирования отдельных битов входного сообщения R_{i-1}. Порядок следования битов $E(R_{i-1})$ также задан в самом алгоритме DES и не может быть изменен (таблица 2.4).

Таблица 2.4 – Функция расширения E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Полученный результат складывается по модулю 2 с 48 битовым ключом K_i. После этого полученный 48 битовый вектор разделяется на 8

блоков по 6 битов в каждом. Каждый блок поступает на свою S-матрицу (таблица 2.5).

Таблица 2.5

		Номер столбца																
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	
Номер строки	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7	S_1
	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8	
	2	4	1	4	8	13	6	2	11	15	12	9	7	3	10	5	0	
	3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13	
	0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10	S_2
	1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5	
	2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15	
	3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9	
	0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8	S_3
	1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1	
	2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7	
	3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12	
	0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15	S_4
	1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9	
	2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4	
	3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14	
	0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9	S_5
	1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6	
	2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14	
	3	11	8	12	7	1	14	2	13	6	5	0	9	10	4	5	3	
	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11	S_6
	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8	
	2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	1	6	
	3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13	
	0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1	S_7
	1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6	
	2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2	
	3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12	
	0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7	S_8
	1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2	
	2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8	
	3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11	

Преобразование внутри S-матрицы осуществляется по следующему правилу. Первый и последний биты входного значения для матрицы S_j (j=1..8) представляют 2-битовое двоичное число, задающее одну из четырех подстановок, определяемых четырьмя строками таблицы для S_j. Остальные четыре бита определяют столбец. Десятичное значение, находящееся на пересечении заданных таким образом столбца и строки, в 4-битовом представлении будет результатом. Например, если на вход S_j поступает значение 011001, оно задает строку 01 (в десятичном

представлении - 1) и столбец 1100 (в десятичном представлении – 12). На пересечении строки 1 и столбца 12 находится значение 9, которое в двоичном представлении даст значение 1001.

После прохождения S-матриц полученная 32-битовая последовательность подвергается перестановке P, функция которой представлена в таблице 2.6.

Таблица 2.6 - Функция P перестановки битов

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Вычисление ключей

На каждой i-ой итерации алгоритма DES используется текущее значение ключа K_i (48 бит), получаемое из исходного 64-битового ключа по следующему правилу, представленному на рисунке 2.6.

Первоначально пользователи выбирают 56 случайных битов. Восемь битов, находящихся в позициях кратных 8 (8, 16, …, 64), добавляются таким образом, чтобы каждый байт содержал нечетное число единиц, что используется для обнаружения ошибок при обмене и хранении ключей. Выбранные пользователем 56 битов ключа подвергаются перестановке G (таблица 2.7). Эта перестановка определяется двумя блоками C_0 и D_0 по 28 бит в каждом (соответственно верхняя и нижняя половины таблицы). Следующие значения C_i и D_i (i=1..16) получаются циклическими сдвигами на основании таблицы 2.8.

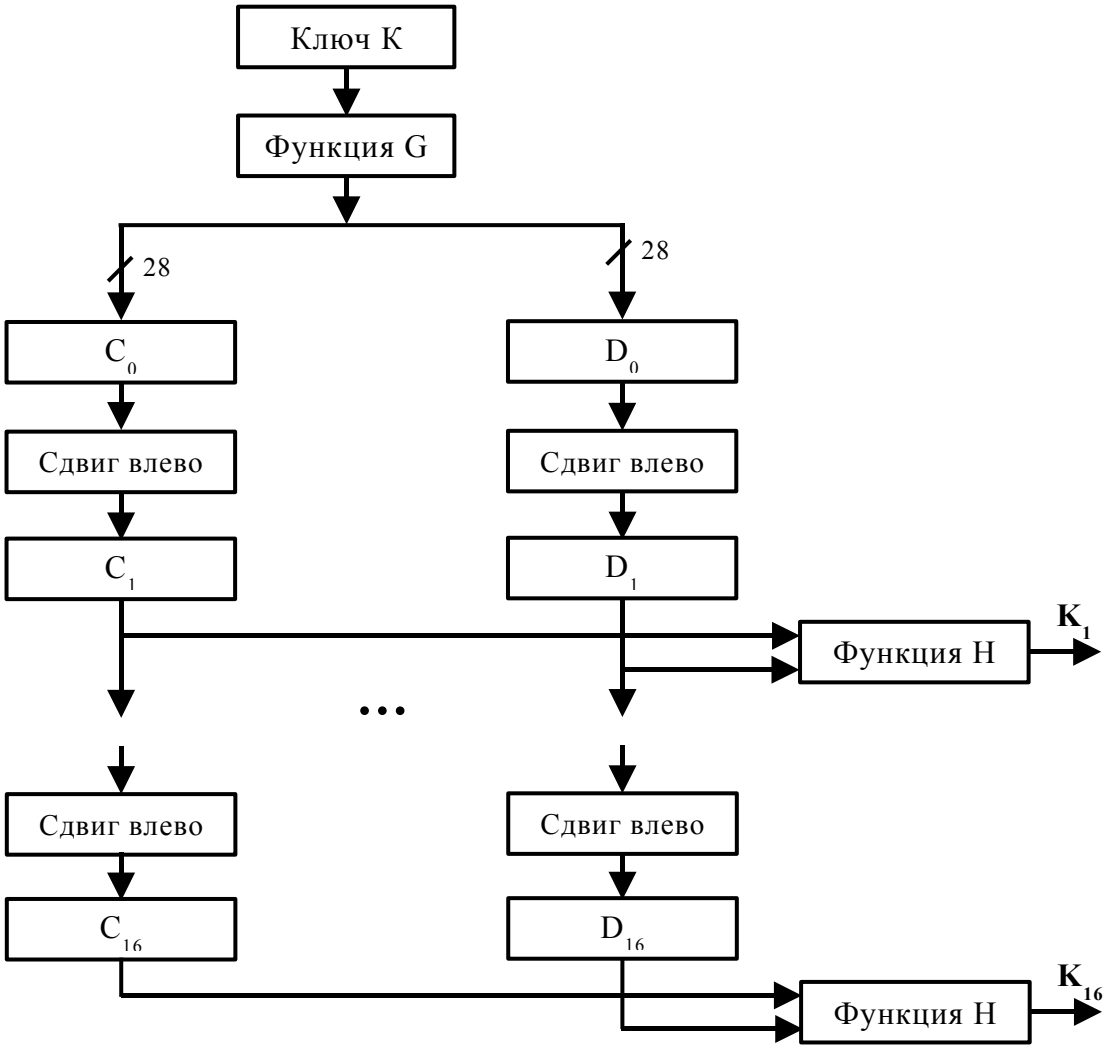

Рисунок 2.6

Таблица 2.7 – Функция G

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Таблица 2.8

i	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Количество сдвигов	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

Определение ключа K_i (i=1..16) определяется на основании функции H от последовательности C_iD_i согласно таблице 2.9.

Таблица 2.9

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Примечания: Со времени принятия DES в качестве стандарта шифрования данных прошло достаточно много времени, параметры, казавшиеся ранее вполне приемлемыми, в настоящее время уже не устраивают. Таким недостатком стало ограниченное по современным меркам количество ключей: длина ключа в 56 бит дает соответственно общее количество ключей – 2^{56}, что позволяет возможность их полного перебора. Ограниченная длина ключа привела сначала к использованию «тройного» DES в различных модификациях [39]:

–с двумя ключами, в этом случае функцию преобразования можно представить в виде последовательности операция шифрования-дешифрования-шифрования

$$C = E_{k1}[D_{k2}[E_{k1}[M]]]$$

длина ключа в этом случае составляет 112 бит.

–с тремя ключами, в этом случае общая длина используемого ключа равна 168 бит, а процесс преобразования может быть описан следующей формулой

$$C = E_{k3}[D_{k2}[E_{k1}[M]]].$$

Для преодоления этой проблемы в 1997 году был объявлен конкурс на разработку нового стандарта шифрования. 2 октября 2000 г. Национальный институт стандартов и технологий (НИСТ) США объявил победителем конкурса AES, этот алгоритм и стал новым стандартом.

Алгоритм AES представляет собой итеративный блочный шифр, имеющий переменную длину блоков и различные длины ключей. Длины ключей и блоков могут независимо друг от друга равняться 128, 192 или 256 бит. [2]

2.3.2 Режимы использования DES

Алгоритм DES может использоваться в четырех основных режимах:
–электронная кодовая книга (ECB – electronic code book);
–сцепление блоков (CBC – cipher block chaining);
–обратная связь по шифртексту (CFB – cipher feed back);
–обратная связь по выходу (OFB – output feed back).

Электронная кодовая книга.

В данном режиме исходное сообщение разбивается на 64-битовые блоки $M=M_1M_2…M_n$. Каждый такой блок шифруется независимо от остальных блоков, однако для всех блоков используется один и тот же ключ шифрования (рисунок 2.7).

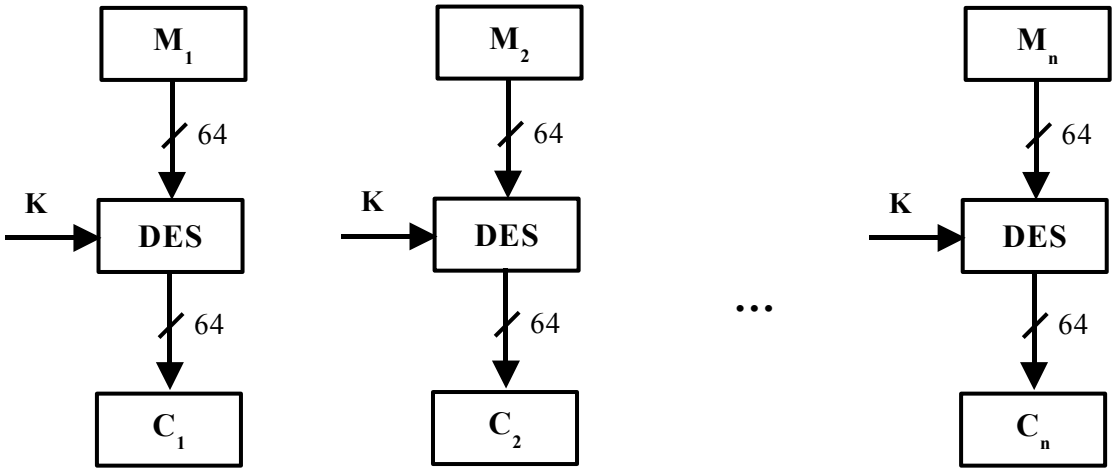

Рисунок 2.7

Основное достоинство данного режима – простота реализации, недостаток – слабая криптостойкость. Поскольку размер блока жестко

фиксирован и возможны случаи повторения в сообщении одних и тех же блоков, в этом случае возможно проведение криптоанализа «со словарем».

Сцепление блоков

Как и в предыдущем режиме исходное сообщение разбивается на 64-битовые блоки: $M=M_1M_2\ldots M_n$. Первый блок складывается по модулю 2 с начальным 64-битовым вектором V, который хранится в секрете. Полученная сумма шифруется с использованием общего секретного ключа K. Полученный 64-битовый блок шифртекста C_1 складывается по модулю 2 со вторым блоком текста, результат также шифруется с помощью ключа K и получается второй блок шифртекста C_2.

Данная процедура будет повторяться до тех пор. пока не будут обработаны все блоки исходного сообщения M (рисунок 2.8).

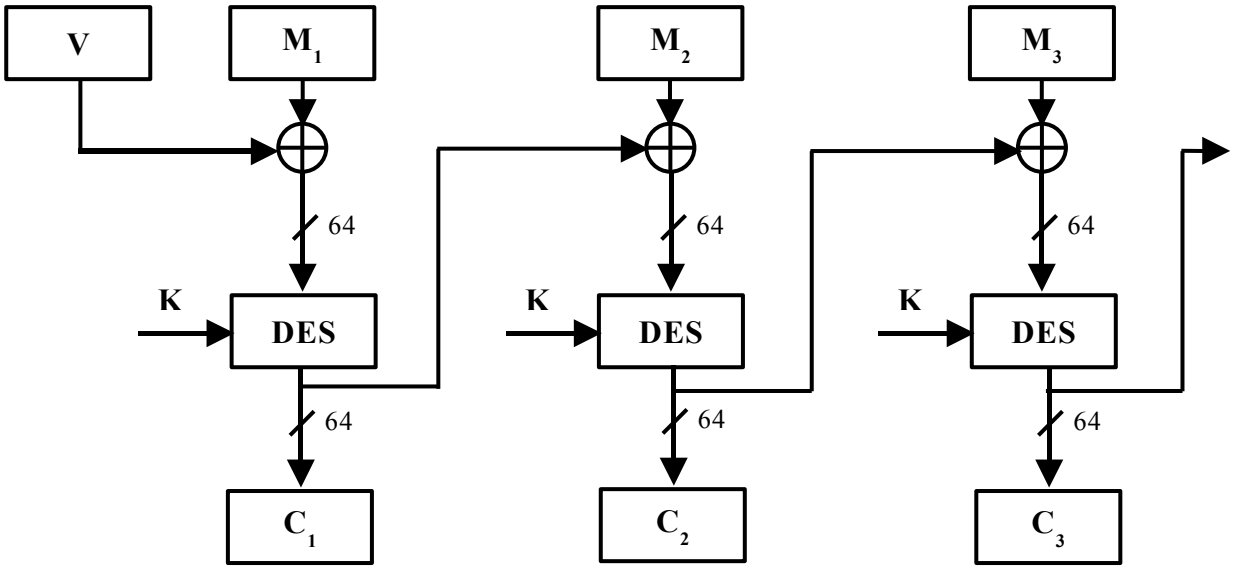

Рисунок 2.8

Последний 64-битовый блок шифртекста можно рассматривать как функцию секретного ключа, начального вектора и каждого бита исходного сообщения. Этот блок шифртекста называется кодом аутентификации сообщения, т.е. проверка данного кода позволяет судить об истинности сообщения.

Обратная связь по шифртексту

В отличие от двух предыдущих режимов размер блока исходного сообщения в данном режиме жестко не фиксирован и может лежать в пределах от 1 до 64 бит (рисунок 2.9).

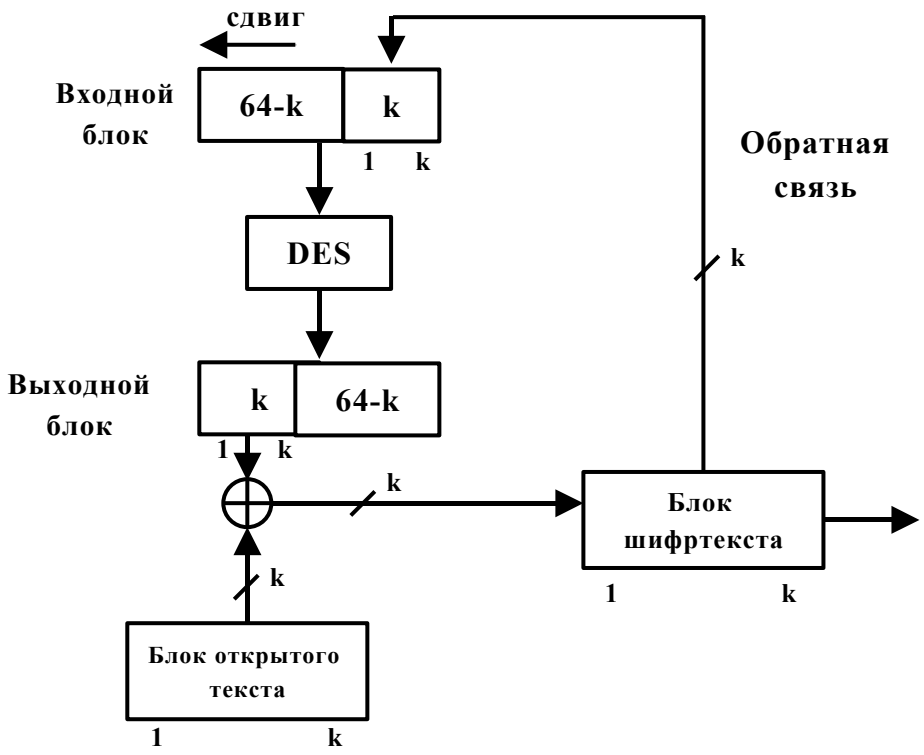

Рисунок 2.9

Входной блок, который представляет собой 64-битовый сдвиговый регистр, вначале содержит вектор инициализации.

Пусть после разбиения исходного сообщения мы имеем n блоков по k бит каждый: $M=M_1M_2\ldots M_n$. Тогда для любого i=1..n блок шифртекста

$$C_i = M_i \oplus P_{i-1},$$

где P_{i-1} – k старших битов предыдущего зашифрованного блока.

Обновление сдвигового регистра осуществляется путем удаления его старших k битов и записи C_i в регистр.

Обратная связь по выходу

Этот режим аналогичен предыдущему, отличие заключается в том, что на вход сдвигового регистра подается не блок шифртекста, а значение выходного блока P_i, которое дописывается справа после сбрасывания k старших битов регистра (рисунок 2.10).

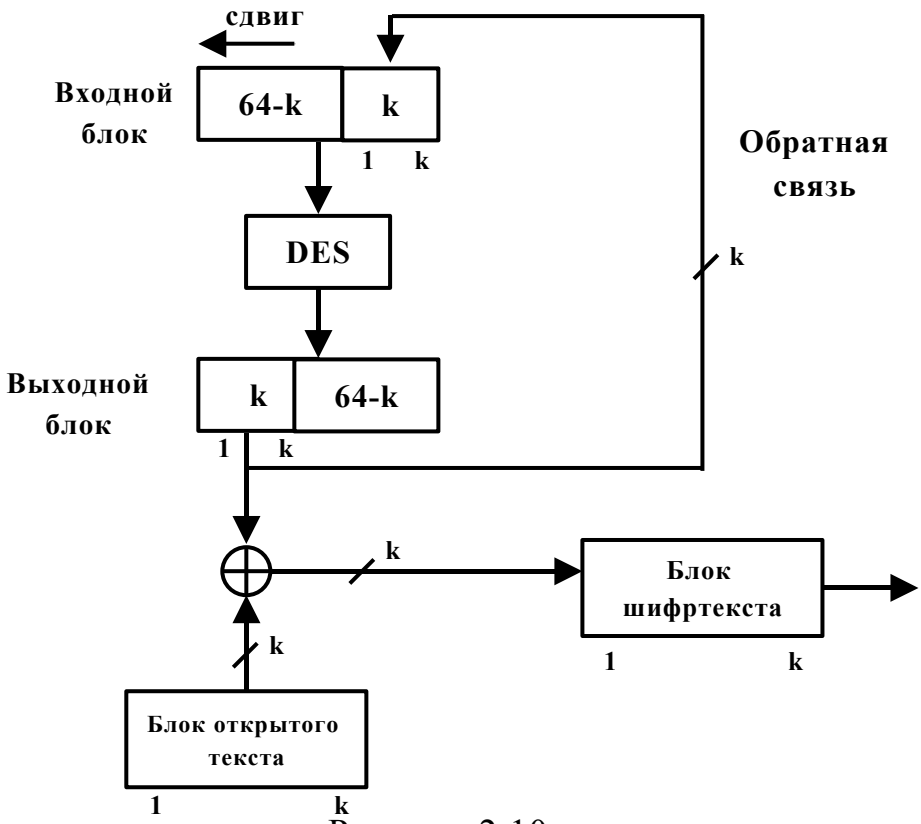

Рисунок 2.10

Для $M=M_1M_2\ldots M_n$ блок шифртекста на каждом шаге i=1..16 вычисляется как

$$C_i = M_i \oplus P_i,$$

где P_i – k старших битов операции DES.

2.3.3 Отечественный стандарт шифрования данных ГОСТ 28147-89

В РФ предусмотрен единый криптографический алгоритм защиты данных в системах обработки информации, в сетях ЭВМ, отдельных вычислительных комплексах или отдельной ЭВМ. Данный алгоритм определяется стандартом ГОСТ 28147-89, который является обязательным для организаций, предприятий и учреждений, применяющих криптографическую защиту данных, хранимых и передаваемых в сетях ЭВМ, в отдельных вычислительных комплексах и ЭВМ. [6, 21, 28]

Этот алгоритм может иметь аппаратную или программную реализацию, удовлетворяет криптографическим требованиям и не накладывает ограничений на степень секретности защищаемой информации.

Алгоритм относится к блочным криптографическим алгоритмам, размер блока – 64 бита. Размер используемой ключевой последовательности – 256 бит.

Введем некоторые обозначения, которые потребуются для описания алгоритма.

$\oplus$ – операция побитового сложения по модулю 2.

$\boxplus$ – операция сложения двух 32-битовых двоичных чисел.

Два целых числа a, b, где $0 \le a,b \le 2^{32} - 1$,

$$a=(a_{32}a_{31}\ldots a_2a_1),\ b=(b_{32}b_{31}\ldots b_2b_1),$$

представлены в двоичном виде, т.е.

$$a = a_{32} \cdot 2^{31} + a_{31} \cdot 2^{30} + \ldots + a_2 \cdot 2^1 + a_1 \cdot 2^0,$$

$$b = b_{32} \cdot 2^{31} + b_{31} \cdot 2^{30} + \ldots + b_2 \cdot 2^1 + b_1 \cdot 2^0,$$

суммируются по модулю 2^{32} (операция $\boxplus$) по следующему правилу:

$$a \boxplus b = a + b, \text{ если } a + b < 2^{32},$$

$$a \boxplus b = a + b - 2^{32}, \text{ если } a + b \ge 2^{32}.$$

ГОСТ 28147-89 предусматривает четыре режима работы:

–шифрование данных в режиме простой замены;

–шифрование данных в режиме гаммирования;

–шифрование данных в режиме гаммирования с обратной связью;

–выработка имитовставки.

Мы рассмотрим работу алгоритма в первом режиме, который является базовым для остальных режимов шифрования.

Исходный текст разбивается на 64-битовые блоки, первые 32 бита заносятся в регистр N_1, а оставшиеся 32 бита – в регистр N_2. Один такт работы состоит в следующем (рисунок 2.11):

1) производится суммирование очередного ключа из ключевого запоминающего устройства (КЗУ) с содержимым регистра N_1 по модулю 2^{32};

2) результат суммирования преобразуется в блоке подстановки: 32-битовый вектор разбивается на восемь последовательно идущих 4-битовых векторов, каждый из которых преобразуется в новый 4-битовй вектор соответствующим узлом замены, затем 4-битовые выходные векторы снова последовательно объединяются в 32-битовый вектор;

3) полученный вектор циклически сдвигается на 11 разрядов в сторону старших разрядов;

4) результат сдвига складывается поразрядно по модулю 2 с содержимым регистра N_2, при этом содержимое регистра N_1 помещается в N_2 (в последнем 32-м цикле результат суммирования заносится в регистр N_2, а значение регистра N_1 сохраняется старое).

5) после 32-го цикла работы в регистрах N_1 и N_2 будет находиться блок шифрования данных, соответствующий исходному блоку открытых данных.

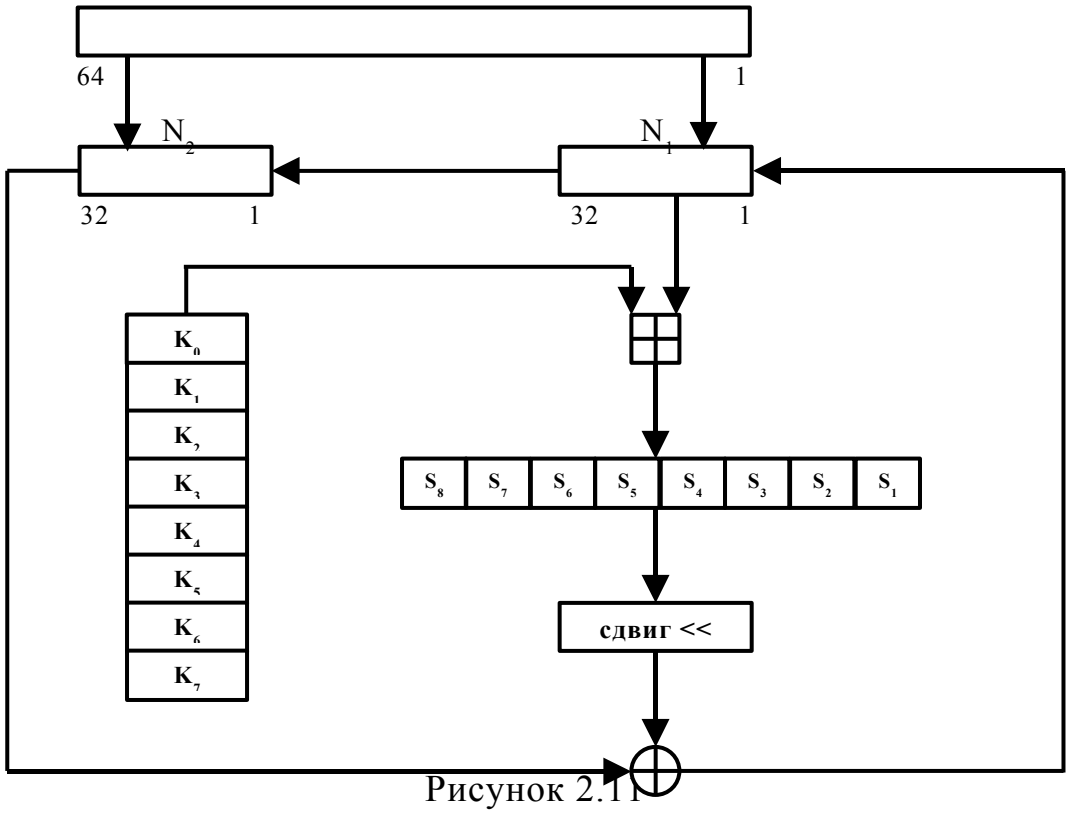

Рисунок 2.11

Как видно из схемы, при каждом применении процедуры шифрования используется своя ключевая последовательность, которая формируется следующим образом. Из 256 бит исходного ключа К формируется 8 векторов по 32 бита в каждом, которые хранятся в КЗУ.

$$K= K_7K_6K_5K_4K_3K_2K_1K_0.$$

При шифровании ключи из КЗУ считываются сначала три раза в прямом порядке (с нулевого по седьмой), а затем один раз - в обратном порядке.

Помимо режима простой замены в алгоритме ГОСТ определены следующие режимы:

–при шифровании данных в режиме гаммирования схема, представленная на рисунке 2.11, с небольшими изменениями используется для выработки 64-битовой гаммы, которая складывается побитово по модулю два с исходным текстом;

–при шифровании данных в режиме гаммирования с обратной связью уже зашифрованный текст используется для выработки последующих блоков гаммы. В обоих этих режимах в регистры N_1 и N_2 первоначально вводится начальная последовательность, от которой будет зависеть используемая в дальнейшем гамма, поэтому при одном и том же ключе, но разной начальной последовательности, будет получен различный шифртекст;

–режим выработки имитовставки используется не только для шифрования, но и проверки целостности сообщения: вначале для исходного текста вырабатывается имитовставка, представляющая дополнительный блок бит, а затем уже данные шифруются. Процесс выработки имитовставки заключается в последовательном применении преобразования, соответствующего 16-ти циклам простой замены с прибавлением по модулю два после каждых 16-ти циклов следующих блоков открытого текста. В заключении из полученных 64 бит выбирается d бит, которые и добавляются к зашифрованному сообщению в качестве имитовставки. Данные шифруется на основе любого из вышеперечисленных режимов шифрования.

2.3.4 Сравнение алгоритмов DES и ГОСТ 28147-89

Рассмотрим основные отличия двух блочных алгоритмов шифрования DES и ГОСТ 28147-89.

Во-первых, длина ключа в алгоритме DES равна 56 битам, из которых на каждом этапе формируются 16 ключей по 48 бит в каждом. В алгоритме ГОСТ длина ключевой последовательности составляет 256 бит, из которого формируются 8 32-битных ключа.

Во-вторых, несмотря на то, что размер блока в обоих алгоритмах одинаков и равен 64 битам, в алгоритме ГОСТ используется 32 цикла преобразования, а в DES – только 16.

В-третьих, функции замены S в DES являются жестко фиксированными, в алгоритме ГОСТ они никак не определены, что вызывает много проблем при реализации алгоритма, поскольку неизвестно, имеются ли среди возможных узлов замены такие, которые снижают стойкость шифрования информации.

В-четвертых, в алгоритме DES данные подвергаются начальной и конечной перестановке, которая отсутствует в отечественном стандарте.

Оба алгоритма являются стойкими алгоритмами шифрования, в том смысле, что неизвестно таких методов расшифрования, которые кардинально отличались бы по трудоемкости от метода тотального перебора ключа. Однако основной недостаток DES заключается в слишком коротком по современным меркам ключе, о чем уже было сказано выше.

2.4 Асимметричные криптосистемы

2.4.1 Характеристика асимметричных криптосистем

Появление криптографии с открытым ключом (или асимметричной криптографии) стало революционным во всей истории криптографии. Существующие до этого времени криптографические системы базировались в основном на использовании элементарных средств замены и перестановки. Появление электромеханических шифровальных машин, а затем и компьютеров значительно продвинули вперед криптографию как науку, однако вначале по-прежнему использовали все те же традиционные методы в основе криптографических алгоритмов. [1, 3, 16, 39]

Асимметричные криптосистемы радикально изменили традиционные подходы к шифрованию. Прежде всего, стали использоваться математические функции, отличные от перестановок и подстановок. Но самое главное, криптографические алгоритмы с открытым ключом предполагают наличие двух разных ключей для шифрования и расшифрования данных, что позволяет обеспечить не только конфиденциальность, но и целостность данных.

Идея применения асимметричной криптографии возникла из попыток найти решение из двух наиболее сложных проблем, возникающих при использовании традиционного шифрования. Первой проблемой является распределение ключей: при традиционном шифровании требуется, чтобы обе участвующие стороны уже имели общий секретный ключ либо использовали услуги третьей стороны – центра распределения ключей. В последнем случае независимо от стойкости самой криптосистемы, ключ может быть скомпрометирован. Второй проблемой является проблема установления подлинности отправляющей и принимающей стороны; в случае симметричной криптосистемы подлинность участников связи обеспечивается знанием общего секретного ключа, в случае, если ключ будет доступен злоумышленнику, то он также будет считаться подлинным участником.

При асимметричном шифровании для шифрования и расшифровывания используются разные ключи, и знание одного из них не дает практической возможности определить второй. Поэтому ключ для шифрования может быть сделан общедоступным без потери стойкости шифра, если ключ для расшифровывания сохраняется в секрете, например, генерируется и хранится только получателем информации.

У. Диффи и М. Хеллман сформулировали требования, выполнение которых обеспечит безопасность асимметричной криптосистемы:

1) вычисление пары ключей (открытого и закрытого) должно быть простым;

2) отправитель, зная открытый ключ k_o, легко вычисляет криптограмму

$$C=E_{ko}(M)$$

3) получатель, используя закрытый ключ k_c и криптограмму C, легко восстанавливает исходное сообщение:

$$M=D_{kc}(C)$$

4) противник, зная открытый ключ, при попытке вычислить секретный ключ наталкивается на непреодолимую вычислительную проблему;

5) противник, зная открытый ключ и криптограмму, при попытке восстановить исходное сообщение наталкивается на непреодолимую вычислительную проблему.

Общую схему функционирования асимметричной криптосистемы можно представить следующим образом (рисунок 2.12).

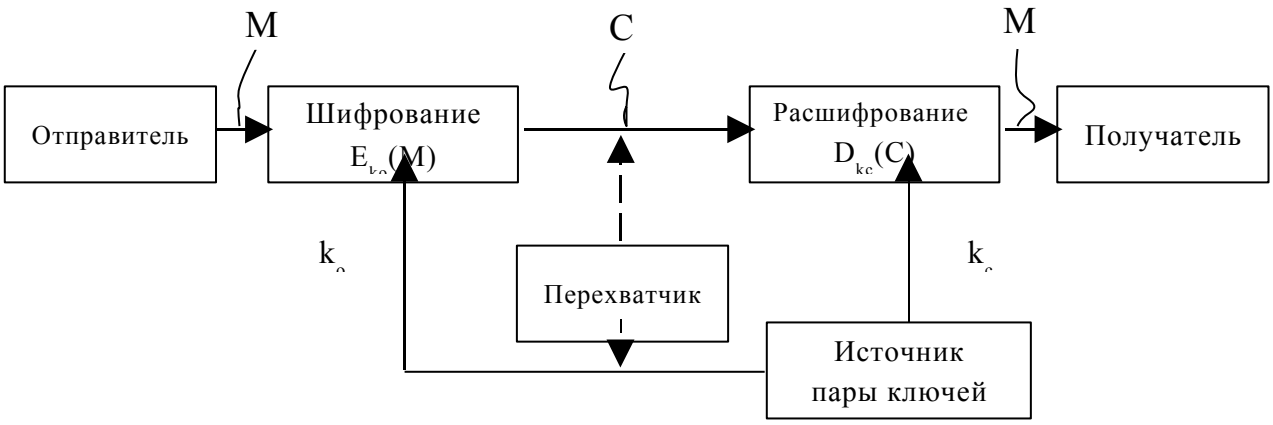

Рисунок 2.12

2.4.2 Однонаправленные функции

Концепция асимметричных криптосистем основана на применении однонаправленных функций. Мы ограничимся неформальным определением данного понятия. [3] *Однонаправленной* называется функция $F : X \rightarrow Y$, обладающая двумя свойствами:

а) существует полиномиальный алгоритм вычисления значений F(x);

б) не существует полиномиального алгоритма инвертирования функции F (т.е. решения уравнения F(x) = y относительно x).

В криптографии широко применяются следующие односторонние функции.

Разложение большого числа на простые сомножители (каноническое разложение):

$$N = P \cdot Q,$$

где

P, Q – простые числа.

По современным оценкам теории чисел при целом $N \approx 2^{664}$ и приблизительно равных сомножителях потребуется около 10^{23} операций, т.е. эта задача практически неразрешима на современных ЭВМ.

Задача дискретного логарифмирования:

$$A^x \bmod N = y$$

Алгоритм вычисления дискретного логарифма пока не найден, поэтому данная функция считается односторонней (однако не доказано, что эффективного алгоритма не существует). При целых $A \approx 2^{664}$ и $N \approx 2^{664}$ решение задачи дискретного логарифмирования (нахождение показателя степени x для известного y) потребует около 10^{26} операций, т.е. эта задача имеет еще большую вычислительную сложность, чем предыдущая.

2.4.3 Криптосистема RSA

Первой и наиболее известной криптографической системой с открытым ключом была предложенная в 1978 году система RSA. Ее название происходит от первых букв фамилий авторов Rivest, Shamir и Adleman. Она основана на трудности разложения очень больших целых чисел на простые сомножители.

Процедуру шифрования данных алгоритмом RSA (как и любым другим асимметричным алгоритмом) можно разбить на два этапа: определение пары ключей (секретного и открытого) и непосредственно шифрования данных. Открытый текст шифруется блоками, каждый из которых содержит двоичное значение, меньшее некоторого заданного числа n. На практике длина блока выбирается равной 2^k битам, где $2^k < n \leq 2^{k+1}$.

I Определение ключей.

1. Получатель выбирает два очень больших простых числа P и Q и вычисляет их произведение

$$N = P \cdot Q$$

2. Получатель вычисляет функцию Эйлера по формуле

$$\varphi(N) = (P - 1) \cdot (Q - 1)$$

3. Затем он выбирает случайное целое число K_o, исходя из следующих условий

$$1 < K_o \le \varphi(N) \quad \text{и} \quad (K_o, \varphi(N)) = 1$$

K_o и N объявляются открытыми ключами и пересылаются отправителю.

4. Закрытый ключ K_c находится из решения линейного сравнения

$$K_o \cdot K_c = 1 \bmod \varphi(N)$$

II Шифрование/расшифрование данных.

1. Если M – сообщение (длина которого, определяемая по значению выражаемого им целого числа, должна принадлежать интервалу (1, N)), то отправитель вычисляет криптограмму по следующей формуле

$$C = M^{Ko} \bmod N$$

2. Получатель сообщения расшифровывает его следующим образом

$$M = C^{Kc} \bmod N$$

Пример. Зашифровать сообщение M=15, используя следующие входные данные: P=23, Q=37, K_o=47, а затем расшифровать полученную криптограмму.

I Определяем ключи.

1. Вычисляем модуль N

$$N = P \cdot Q = 23 \cdot 37 = 851$$

2. Рассчитываем функцию Эйлера от модуля N

$$\varphi(N) = (P-1)\cdot(Q-1) = (23-1)\cdot(37-1) = 22\cdot 36 = 792$$

3. Значения K_o=47 и N=851 объявляются открытыми ключами и пересылаются отправителю.

4. Находим закрытый ключ K_c, решив следующее линейное сравнение

$$47\cdot K_c = 1 \bmod 792$$

а) находим неполные частные с использованием алгоритма Евклида для чисел 792 и 19.

$$792 = 47\cdot 16 + 40 \qquad q_1 = 16$$
$$47 = 40\cdot 1 + 7 \qquad q_2 = 1$$
$$40 = 7\cdot 5 + 5 \qquad q_3 = 5$$
$$7 = 5\cdot 1 + 2 \qquad q_4 = 1$$
$$5 = 2\cdot 2 + 1 \qquad q_5 = 2$$
$$2 = 1\cdot 2 \qquad q_6 = 2$$

n=6

б) P_0=1, P_1=q_1.

в)

$$P_2 = q_2\cdot P_1 + P_0 \qquad P_2 = 1\cdot 16 + 1 = 17$$
$$P_3 = q_3\cdot P_2 + P_1 \qquad P_3 = 5\cdot 17 + 16 = 101$$
$$P_4 = q_4\cdot P_3 + P_2 \qquad P_4 = 1\cdot 101 + 17 = 118$$
$$P_5 = q_5\cdot P_4 + P_3 \qquad P_5 = 2\cdot 118 + 101 = 337$$

г) $K_c = (-1)^5\cdot P_5\cdot b \bmod m = (-1)\cdot 337 \bmod 792 = 455 \bmod 792$

II Шифруем сообщение.

1. M=15 (M<N)

$$C = M^{Ko} \bmod N = 15^{47} \bmod 851 = 500$$

3. Проверим, расшифровав сообщение

$$M = C^{Kc} \bmod N = 500^{455} \bmod 851 = 15$$

3.3.3 Криптосистема Эль Гамаля (El Gamal)

Еще одной асимметричной криптосистемой является система, которую предложил в 1985 году Эль Гамаль. [30]

I Определение ключей.

1. С целью генерации пары ключей вначале выбирается большое простое число P и большое целое число G. Эти значения являются открытыми.

2. Выбирается случайное число X (X<P), которое является секретным ключом.

3. Вычисляется

$$Y = G^X \bmod P$$

Y является открытым ключом.

Сообщения представляются целыми числами M из интервала (1, P).

II Шифрование/расшифрование данных.

1. Для шифрования сообщения M (длина которого (1, P)), выбирается случайное число K, 1<K<P-1, (K,P-1)=1.

2. Отправитель вычисляет следующие числа

$$a = G^K \bmod P$$

$$b = Y^K \cdot M \bmod P$$

Пара чисел (a,b) является криптограммой исходного сообщения M.

3. Получатель расшифровывает криптограмму следующим образом:

$$M = b / a^X \bmod P = b \cdot a^{-X} \bmod P$$

Доказательство:

$$b / a^X = Y^K \cdot M / a^X = G^{XK} \cdot M / G^{KX} \bmod P = M \bmod P$$

Пример. Зашифровать сообщение M=40, используя следующие входные данные: P=157, G=83, X=34 (секретный ключ), а затем расшифровать полученную криптограмму.

I Определение ключей.

1. Значения P и G, которые являются открытыми, уже даны в условии задачи, так же как и X.

2. Вычислим открытый ключ

$$Y = G^X \bmod P = 83^{34} \bmod 157 = 25$$

II Шифрование данных.

1. Исходя из следующих условий, 1<K<P-1, (K,P-1)=1, выберем K=29.

2. Вычисляем криптограмму

$$a = G^K \bmod P = 83^{29} \bmod 157 = 84$$

$$b = Y^K \cdot M \bmod P = 25^{29} \cdot 40 = 110$$

Пара чисел (84,110) отправляется получателю.

3. Расшифруем криптограмму:

$$M = b / a^X \bmod P = b \cdot a^{-X} \bmod P$$

Для расшифрования полученного сообщения предварительно найдем значение $a^{-x} \bmod P$, решив линейное сравнение

$$a^X \cdot a^{-X} = 1 \bmod P$$

Для удобства сделаем следующее обозначение

$$y = a^{-X}$$

Следовательно, наше линейное сравнение будет выглядеть следующим образом

$$84^{34} \cdot y = 1 \bmod 157$$

$$42 \cdot y = 1 \bmod 157$$

а) находим неполные частные с использованием алгоритма Евклида для чисел 157 и 42.

$157 = 42 \cdot 3 + 31$	$q_1 = 3$
$42 = 31 \cdot 1 + 11$	$q_2 = 1$
$31 = 11 \cdot 2 + 9$	$q_3 = 2$
$11 = 9 \cdot 1 + 2$	$q_4 = 1$
$9 = 2 \cdot 4 + 1$	$q_5 = 4$
$2 = 1 \cdot 2$	$q_6 = 2$

n=6

б) $P_0=1$, $P_1=q_1$

в)

$P_2 = q_2 \cdot P_1 + P_0$	$P_2 = 1 \cdot 3 + 1 = 4$
$P_3 = q_3 \cdot P_2 + P_1$	$P_3 = 2 \cdot 4 + 3 = 11$
$P_4 = q_4 \cdot P_3 + P_2$	$P_4 = 1 \cdot 11 + 4 = 15$
$P_5 = q_5 \cdot P_4 + P_3$	$P_5 = 4 \cdot 15 + 11 = 71$

г)

$$y = (-1)^5 \cdot P_5 \cdot b \bmod P = (-1) \cdot 71 \bmod 157 = 86 \bmod 157$$

Используя найденное значение, рассчитываем исходное сообщение M

$$M = 110 \cdot 86 \bmod 157 = 40$$

3 Криптографические протоколы

3.1 Общая характеристика

В классической шенноновской модели секретной связи имеются два полностью доверяющих друг другу участника, которым необходимо передавать между собой информацию, не предназначенную для третьих лиц, т.е. конфиденциальную информацию. Защита секретной или конфиденциальной информации является первой задачей криптографии и решается с помощью криптосистем.

Однако бывают случаи, когда в самой пересылаемой информации ничего секретного нет, но требуется гарантия поступления сообщений из достоверного источника в неискаженном виде. Такая гарантия называется обеспечением целостности информации и составляет вторую задачу криптографии. Для обеспечения целостности информации используются специальные алгоритмы – криптографические протоколы. В отличие от криптосистем [3]:

–криптографические протоколы могут быть интерактивными, т.е. подразумевать многораундовый обмен сообщениями между участниками;

–в криптографических протоколах может быть более двух участников;

–считается, что участники протокола не доверяют друг другу.

3.2 Протоколы идентификации и аутентификации

3.2.1 Характеристика протоколов идентификации и аутентификации

С любым объектом вычислительной системы связана некоторая информация, однозначно идентифицирующая его. Данную информацию называют идентификатором объекта. Если

объект имеет некоторый идентификатор, зарегистрированный в сети, то он называется законным (легальным объектом).

Идентификация объекта – функция вычислительной системы, которая выполняется, когда объект пытается войти в систему, сеть.

Если процедура идентификации выполнилась успешно и объект признан легальным, выполняется следующая процедура – аутентификация объекта, т.е. устанавливается, является ли объект именно тем, за кого он себя выдает, другими словами эта процедура называется проверкой подлинности объекта.

Если объект идентифицирован и признан подлинным, устанавливается сфера деятельности объекта и доступные ему ресурсы. Данная процедура называется авторизацией (установкой полномочий).

При защите каналов передачи данных мы имеем дело со взаимным установлением подлинности объектов, связывающихся между собой по линиям связи.

Процедура проверки подлинности обычно выполняется в начале сеанса связи. Цель данной процедуры – обеспечить уверенность в том, что соединение устанавливается с законным объектом и вся информация дойдет до места назначения.

При обмене информацией необходимо выполнить следующие требования защиты:

1)получатель должен быть уверен в подлинности источника данных;

2)получатель должен быть уверен в подлинности передаваемых данных;

3)отправитель должен быть уверен в доставке данных получателю;

4)отправитель должен быть уверен в подлинности доставленных данных.

Если в вычислительной системе выполняются все 4 требования, то гарантируется защита данных при передаче по каналу связи и обеспечивается функция защиты, называемая функцией подтверждения передачи. В этом случае отправитель не может отрицать ни факта посылки сообщения, ни его содержания, а получатель не может отрицать ни факта получения сообщения, ни подлинности его содержания.

Все протоколы идентификации и аутентификации включают двух участников:

1) А – доказывающего – участника, проходящего идентификацию;

2) В –проверяющего – участника, проверяющего аутентичность доказывающего.

Целью протокола является проверка подлинности А. Возможными исходами протокола может быть принятие решения об идентичности доказывающего А либо завершение протокола без принятия такового решения.

Выделяют три категории протоколов идентификации и аутентификации:

–протоколы, основанные на известной обеим сторонам информации (различные пароли, идентификационные номера). Знание этой информации демонстрируется во время выполнения протокола;

–протоколы, использующие физические приборы, с помощью которых проводится идентификация, например, магнитная или интеллектуальная пластиковая карта;

–протоколы биометрической идентификации личности. Данный вид протоколов не относится к криптографическим, однако мы уделим внимание этому вопросу в данном разделе.

3.2.2 Идентификация на основе пароля

Допустим, что в вычислительной системе зарегистрировано n пользователей и каждый пользователь имеет аутентифицирующий объект, который содержит два информационных поля [30]:

ID_i – идентифицирующая информация i-го пользователя (логин);

K_i – аутентифицирующая информация i-го пользователя (например, пароль).

Рассмотрим типовые схемы идентификации и аутентификации пользователей при входе в систему.

Схема 1

Вычислительная система содержит объект-эталон, служащий для идентификации и аутентификации пользователей.

Структура объекта-эталона

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1	E_1
2	ID_2	E_2
3	ID_3	E_3
…	…	…

$E_i = F(ID_i, K_i)$, причем F - однонаправленная функция.

Протокол идентификации и аутентификации:

Пользователь предъявляет свой идентификатор ID.

1. Если ID не совпадает ни с одним ID_i, зарегистрированным в вычислительной системе, то идентификация отвергается, иначе считается, что пользователь идентифицировался.

2. У пользователя запрашивается идентификатор К.

3. Субъект идентификации вычисляет $Y = F(ID_i, K_i)$

4. Вычисленное значение сравнивается с E_i. Если $Y = E_i$, то считается, что объект аутентифицирован и информация об этом передается другим модулям вычислительной системы, иначе пользователь не допускается к работе.

Схема 2

Структура модифицированного объекта-эталона

Номер пользователя	Информация для идентификации	Информация для аутентификации
1	ID_1S_1	E_1
2	ID_1S_2	E_2
3	ID_1S_3	E_3
…	…	…

где S_i - случайный вектор («соль»), который задается при создании идентификатора пользователя;

$E_i = F(S_i, K_i)$.

Протокол идентификации и аутентификации:

1. Пользователь предъявляет свой идентификатор ID.

2. Если ID не совпадает ни с одним ID_i, зарегистрированным в вычислительной системе, то идентификация отвергается, иначе считается, что пользователь идентифицировался.

3. По значению ID_i выделяется вектор S_i.

4. Субъект идентификации вычисляет $Y = F(S_i, K_i)$.

5. Субъект аутентификации сравнивает Y и E_i. Если $Y = E_i$, то считается, что аутентификация прошла успешно, иначе пользователь не допускается к работе.

Исследования показали, что в системе необязательно знать сами пароли, достаточно, чтобы компьютер мог отличить правильные пароли от неправильных. Это легко можно достичь с помощью односторонних функций. При этом хранятся сами значения односторонних функций, а не сами пароли, как было показано ранее. Следовательно, если компьютер не хранит таблицу паролей, снижается угроза ее похищения, т.к. по значению односторонней функции не удастся получить исходные аргументы.

Однако таблицы, рассмотренные выше (схема 1), уязвимы. Имея запас времени, можно составить список из наиболее часто встречающихся паролей, обработать их односторонней функцией и сохранить результат. Похитив таблицу паролей, можно сравнить ее с полученным результатом и найти совпадения.

"Соль" – способ затруднить вскрытие с помощью словаря. "Соль" представляет собой случайную строку, добавляемую перед обработкой данных односторонней функцией. Использование достаточно большого числа возможных значений "соли" практически устраняет возможность вскрытия с помощью словаря. Например, большинство UNIX систем использует для «соли» 12 бит.

Правила составления паролей:

1) основным требованием при составлении пароля является ограничение на его минимальную длину. Современные источники рекомендуют устанавливать значение не менее 8 символов. В противном случае, пароль может быть подвержен атаке тотального перебора;

2) наличие в пароле разных групп символов (верхний, нижний регистры, цифры), а также пароль не должен быть словом

/ словосочетанием. В этом случае к паролю не сможет быть применена атака подбора со словарем;

3) для того, чтобы увеличить неопределенность используемых паролей, а также сохранить такое важное свойство как возможность запоминания человеком, можно использовать парольные фразы, когда вместо короткого слова используется целое предложение.

3.2.3 Взаимная проверка подлинности пользователей

Для проверки подлинности применяют следующие способы:

–механизм запроса-ответа;

–механизм отметки времени.

Механизм запроса-ответа состоит в следующем. Если пользователь А хочет быть уверенным, что сообщения, получаемые им от имени пользователя В, не являются ложными, он включает в посылаемое для В сообщение непредсказуемый элемент. При ответе пользователь В должен выполнить над этим элементом некоторую операцию. Это невозможно выполнить заранее, поскольку неизвестно, какое значение будет иметь непредсказуемый элемент. Получив ответ с результатом операции, пользователь А принимает решение о подлинности пользователя В. К недостаткам данного метода можно отнести возможность установления закономерности между запросом и ответом.

Механизм отметки времени подразумевает регистрацию времени для каждого сообщения. В этом случае каждый пользователь сети может определить, насколько «устарело» пришедшее сообщение, и решить не принимать его, поскольку оно может быть ложным.

В обоих случаях для защиты механизма контроля следует применять шифрование, чтобы быть уверенным, что ответ послан не злоумышленником.

Для взаимной проверки подлинности обычно используют процедуру «рукопожатия», в которой стороны признают подлинность друг друга, если докажут, что обладают правильными ключами.

Рассмотрим в качестве примера процедуру рукопожатия для пользователей A и B (в качестве которых могут выступать рабочие станции, сервер и т.п.) [30].

Пользователи A и B имеют общий секретный ключ $К_{AB}$ (рисунок 3.1).

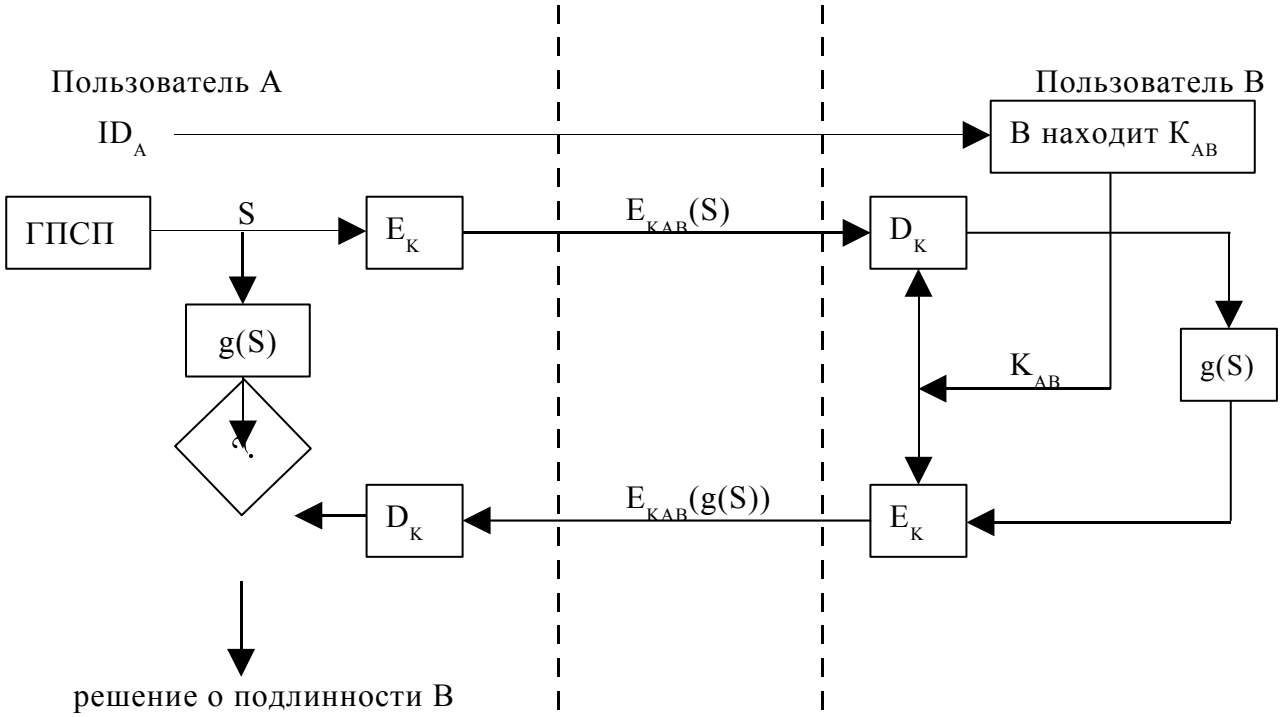

Рисунок 3.1

1. Пользователь A инициирует процедуру рукопожатия, отправляя пользователю B свой идентификатор ID_A в открытой форме.

2. Пользователь B, получив идентификатор ID_A, находит в базе данных секретный ключ $К_{AB}$ и вводит его в свою криптосистему.

3. Пользователь A генерирует случайную последователь ность S и отправляет ее пользователю B в виде криптограммы $E_{KAB}(S)$.

4. Пользователь B расшифровывает криптограмму и раскрывает исходный вид последовательности S.

5. Затем оба пользователя A и B преобразуют последовательность S, используя одностороннюю функцию g(S).

6. Пользователь B шифрует g(S) и отправляет криптограмму пользователю A.

7. Пользователь A расшифровывает криптограмму и сравнивает полученное значение с исходным. Если они равны, то пользователь A признает подлинность B.

Пользователь B таким же образом проверяет подлинность A.

Особенность данного протокола состоит в том, что A и B уже имеют общий секретный ключ.

3.2.4 Протоколы идентификации с нулевой передачей знаний

Недостатком протоколов с фиксированным паролем является то, что доказывающий A передает проверяющему B секретную информацию – пароль. Это впоследствии может привести к тому, что B сможет воспользоваться этой секретной информацией и выдать себя за A. Протоколы с нулевой передачей знаний призваны устранить эту проблему, давая возможность доказывающему продемонстрировать знание пароля, не выдавая о нем никакой информации. Следует отметить, что доказательство имеет вероятностный характер, т.е. утверждение о подлинности пользователя принимается с некоторой вероятностью, значение которой может быть выбрано принимающей доказательство стороной.

Примером такого вида протоколов является протокол FFS (разработан Фейге (Uriel Feige), Фиатом (Amos Fiat) и Шамиром (Adi Shamir) в 1986 году).

Упрощенная схема идентификации Feige-Fiat-Shamir

Перед выдачей закрытых ключей арбитр выбирает случайный модуль n, являющийся произведением двух больших простых чисел. В реальной жизни длина модуля должна быть не менее 512 бит, лучше если она приближается к 1024 битам. n может быть общим у группы пользователей.

Для генерации открытого и секретного ключей арбитр выбирает число v, являющееся квадратичным остатком по модулю n, т.е. чтобы уравнение

$x^2 = v \bmod n$ имело решение и существовало $v^{-1} \bmod n$. Данное число v будет открытым ключом некоторого пользователя A.

Затем вычисляется наименьшее s из условия

$$s = \sqrt{v^{-1}} \bmod n$$

s будет являться секретным ключом пользователя A.

Протокол идентификации:

1. A выбирает случайное $r < n$.

Вычисляет $x=r^2 \bmod n$

Отсылает x пользователю C.

2. C посылает пользователю A случайный бит b.

3. Если b=0, то A посылает C r.

Если b=1, то A посылает C $y = r \cdot s \bmod n$.

4. Если b=0, то C проверяет $x=r^2 \bmod n$

Если b=1, то C проверяет $x = y^2 \cdot v \bmod n$.

В первом случае C убеждается, что A знает значение $\sqrt{x}$, во втором случае - $\sqrt{v^{-1}}$.

Один этап данного протокола называется аккредитацией. A и C повторяют его, пока C не убеждается, что A знает s. Если A не знает s, то он может обмануть C, подобрав r, в случае когда b=0, или A может обмануть C, подобрав r, если C пошлет b=1. Но A не сможет сделать одновременно и то, и другое. Вероятность того, что A удалось обмануть C за t аккредитаций, равна $\frac{1}{2^t}$.

Чтобы протокол корректно работал, A не должен использовать повторно одно и то же значение r, иначе за два этапа C получит ответы A, по которым сможет вычислить значение s.

Параллельная схема идентификации Feige-Fiat-Shamir

Данная схема может повысить число аккредитаций и уменьшить число взаимодействий A и C.

Выбирается n, равное произведению двух простых чисел. Для генерации отрытого и секретного ключей выбирается k различных чисел: v_1, v_2, … v_k, где v_i – является квадратичным остатком по модулю n, т.е. чтобы

$x^2 = v_i \bmod n$ имело решение и существовало $v_i^{-1} \bmod n$, n=1… k.

Затем вычисляется s_i:

$$s_i = \sqrt{v_i^{-1}} \bmod n$$

Строка v_1, v_2, … v_k служит открытым ключом; вектор s_1, s_2, … s_k, является закрытым ключом.

Протокол:

1. A выбирает случайное $r < n$. Вычисляет $x=r^2 \bmod n$

Отсылает x пользователю C.

2. C посылает A k случайных битов b_1, b_2, … b_k

3. A вычисляет $y = r \cdot (s_1^{b1} \cdot s_2^{b2} \cdot \ldots s_k^{bk}) \bmod n \bmod n$ и отсылает A.

4. C проверяет, что $x = y^2 (v_1^{b1} \cdot v_2^{b2} \cdot \ldots v_k^{bk}) \bmod n$

A и C повторяют этот протокол t раз, пока C не убеждается, что A знает секретный ключ.

Вероятность того, что A удастся обмануть C t раз равна $\frac{1}{2^{kt}}$. Авторы предлагают использовать значения k=5 и t=4.

3.3 Электронная цифровая подпись

3.3.1 Целостность данных и аутентификация сообщений

Средства контроля целостности программ и файлов данных, хранимых в вычислительной системе, должны обеспечивать защиту от несанкционированного изменения этой информации нарушителем, особенно при передаче ее по каналам связи, когда

возникает проблема аутентификации автора документа, самого документа и отсутствия изменений в этом документе.

Основные виды злоумышленных действий, защита от которых является целью аутентификации, следующие [30]:

–*активный перехват*: нарушитель перехватывает документы (файлы) и изменяет их;

–*маскарад*: абонент С пересылает документ абоненту А от имени абонента В;

–*ренегатство*: абонент А заявляет, что не посылал сообщения абоненту В, хотя на самом деле посылал;

–*подмена*: абонент В изменяет или формирует новый документ и заявляет, что получил его от абонента А;

–*повтор*: абонент С повторяет ранее переданный документ, который абонент А посылал абоненту В.

При обмене обычными печатными документами таких проблем не возникает. В конце обычного письма или документа исполнитель обычно ставит свою подпись. Подобное действие преследует две цели:

получатель имеет возможность убедиться в истинности письма, сличив подпись с имеющимся у него образцом;

личная подпись является юридическим гарантом авторства документа.

Если подделать подпись человека на бумаге весьма непросто, а проверить подлинность такой подписи современными криминалистическими методами – техническая деталь, то с электронным документом дело обстоит иначе, незаметно внести нелегальные изменения в документ может любой пользователь.

Цифровая подпись является одним из наиболее используемых механизмов решения вышеуказанных проблем.

Рассмотрим отличительные особенности обычной и цифровой подписи. При обычной подписи:

–каждая личность использует индивидуальные, только ей присущие характеристики – почерк, давление на ручку и т.п.;

–попытка подделки подписи распознается с помощью графологического анализа;

–подпись и подписываемый документ передаются вместе на одном листе бумаги;

–подпись не зависит от содержания документа, на котором она поставлена;

–копии подписанных документов недействительны, если каждая из этих копий не имеет своей настоящей подписи.

При цифровой подписи:

–каждая личность использует для подписи документов свой уникальный секретный ключ;

–любая попытка подписать документ без знания соответствующего секретного ключа практически не имеет успеха;

–цифровая подпись документа есть функция от содержания этого документа и секретного ключа;

–цифровая подпись может передаваться отдельно от документа;

–копия документа с цифровой подписью не отличается от оригинала.

Для аутентификации информации Диффи и Хеллман в 1976 г. предложили концепцию "цифровой подписи". Она заключается в том, что каждый абонент сети имеет личный секретный ключ, на котором он формирует подпись и известную всем другим проверочную комбинацию, необходимую для проверки подписи (открытый ключ). Цифровая подпись формируется на основе сообщения и секретного ключа отправителя. Любой получатель, имеющий соответствующую проверочную комбинацию, может аутентифицировать сообщение по подписи. При этом знание лишь проверочной комбинации не позволяет подделать подпись. Такие схемы называются асимметричными схемами аутентификации.

Термин "цифровая подпись" используется для методов, позволяющих установить подлинность автора сообщения при возникновении спора относительно авторства этого сообщения. Цифровая подпись используется в вычислительных системах, в которых отсутствует взаимное доверие сторон.

Известны два класса формирования цифровой подписи [1, 7, 8]:

–первый класс способов использует однонаправленные функции. К этому классу относится Российский ГОСТ на цифровую подпись (ГОСТ Р 34.10-2001 (взамен 34.10-94) и хэш –

ГОСТ Р 34.11-94). Он является усложнением алгоритмов цифровой подписи RSA и Эль-Гамаля.

–второй класс использует криптостойкие преобразования, зависящие от секретного ключа. Например, используя алгоритм DES в режиме CFB (обратная связь по шифртексту).

Формирование ЭЦП происходит следующим образом. Отправитель вычисляет хэш-функцию H(M) подписываемого документа (M – текст документа) и получает блок информации m=H(M), характеризующий весь текст M в целом. Затем m шифруется секретным ключом отправителя. Получаемая при этом пара чисел представляет собой ЭЦП данного текста M.

При проверке ЭЦП получатель снова вычисляет H(M), после чего при помощи открытого ключа отправителя проверяет, соответствует ли получаемая подпись вычисленному значению m хэш-функции.

3.3.2 Хэш-функции

Дадим неформальное определение хэш-функции.

Хэш-функцией (или функцией хэширования) называется функция, которая осуществляет сжатие строки чисел произвольного размера в строку чисел фиксированного размера (свертку).

Большинство хэш-функций строится на основе однонаправленной функции.

Выделяют два важных типа криптографических хэш-функций: ключевые и бесключевые. Ключевые хэш-функции называются *кодами аутентификации сообщений* и дают возможность без дополнительных средств гарантировать как правильность источника данных, так и целостность данных в системах с доверяющими друг другу пользователями.

Бесключевые хэш-функции называются кодами обнаружения ошибок и дают возможность с помощью дополнительных средств гарантировать целостность данных. Такие функции могут

применяться в системах как с доверяющими, так и не доверяющими друг другу пользователями.

В данной работе мы ограничимся рассмотрением только второго типа хэш-функций – бесключевых, поскольку в пособии представлен только один аспект применения – в электронной цифровой подписи.

К бесключевым хэш-функциям обычно предъявляются следующие требования:

–однонаправленность;

–устойчивость к коллизиям;

–устойчивость к нахождению второго прообраза.

Данные требования означают соответственно высокую сложность нахождения сообщения с заданным значением свертки; пары сообщений с одинаковыми значениями свертки; второго сообщения с тем же значением свертки для заданного сообщения с известным значением свертки.

Рассмотрим алгоритм, реализующий бесключевую хэш-функцию.

SHA

NIST, вместе с NSA, для стандарта цифровой подписиDSA разработал алгоритм безопасного хэширования (Secure Hash Algorithm, SHA). (Сам стандарт называется Стандарт безопасного хэширования (Secure Hash Standard, SHS), а SHA - это алгоритм, используемый в стандарте.)

SHS определяет алгоритм безопасного хэширования (Secure Hash Algorithm, SHA), необходимый для обеспечения безопасности алгоритма цифровой подписи (Digital Signature Algorithm, DSA). Для любого входного сообщения длиной меньше 264 битов SHA выдает 160-битовый результат, называемый кратким содержанием сообщения. Далее, краткое содержание сообщения становится входом DSA, который вычисляет подпись для сообщения.

SHA называется безопасным, так как он разработан так, чтобы было вычислительно невозможно найти сообщение, соответствующее данному краткому содержанию сообщения или найти два различных сообщения с одинаковым кратким

содержанием сообщения . Любые изменения, произошедшие при передаче сообщения, с очень высокой вероятностью приведут к изменению краткого содержания сообщения, и подпись не пройдет проверку. [42]

Описание алгоритма SHA

Во-первых, сообщение дополняется таким образом, чтобы его длина была кратной 512 битам: сначала добавляется 1, а затем нули так, чтобы длина полученного сообщения была на 64 бита меньше числа, кратного 512, а затем добавляется 64-битовое представление длины оригинального сообщения.

Инициализируются пять 32-битовых:

A = 0x67452301

B = 0xefcdab89

C = 0x98badcfe

D = 0x10325476

E = 0xc3d2elf0

Главный цикл алгоритма обрабатывает сообщение 512-битовыми блоками и продолжается, пока не исчерпаются все блоки сообщения.

Сначала пять переменных копируются в другие переменные: A в a, B в b, C в c, D в d и E в e.

Главный цикл состоит из четырех этапов по 20 операций в каждом. Каждая операция представляет собой нелинейную функцию над тремя из a, b, c, d и e, а затем выполняет сдвиг и сложение. В SHA используется следующий набор нелинейных функций:

$f_t(X,Y,Z) = (X \wedge Y) \vee ((\neg X) \wedge Z)$, для t=0 до 19

$f_t(X,Y,Z) = X \oplus Y \oplus Z$, для t=20 до 39

$f_t(X,Y,Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$, для t=40 до 59

$f_t(X,Y,Z) = X \oplus F \oplus Z$, для t=60 до 79

в алгоритме используются следующие четыре константы:

K_t = 0x5a827999, для t=0 до 19

K_t = 0x6ed9ebal , для t=20 до 39

K_t = 0x8flbbcdc, для t=40 до 59

K_t = 0xca62cld6, для t=60 до 79

Блок сообщения превращается из 16 32-битовых слов (M_0 по M_{15}) в 80 32-битовых слов (W_0 no W_{79}) с помощью следующего алгоритма:

$W_t = M_t$, для t = 0по 15

$W_t = (W_t\text{-}3 \oplus W_{t-8}. \oplus W_{t-14} \oplus W_{t-16}) <<< 1$, для t = 16 по 79

Если *t* - это номер операции (от 1 до 80), W, представляет собой t-ый подблок расширенного сообщения, а <<<s - это циклический сдвиг влево на s битов, то главный цикл выглядит следующим образом:

FOR t = 0 to 79

TEMP = (a <<< 5) $+f_t(b,c,d)$ + e+W_t + K_t

e = d

d = c

c = b <<< 30

b = a

a = TEMP

На рисунке 3.2 показана одна операция SHA.

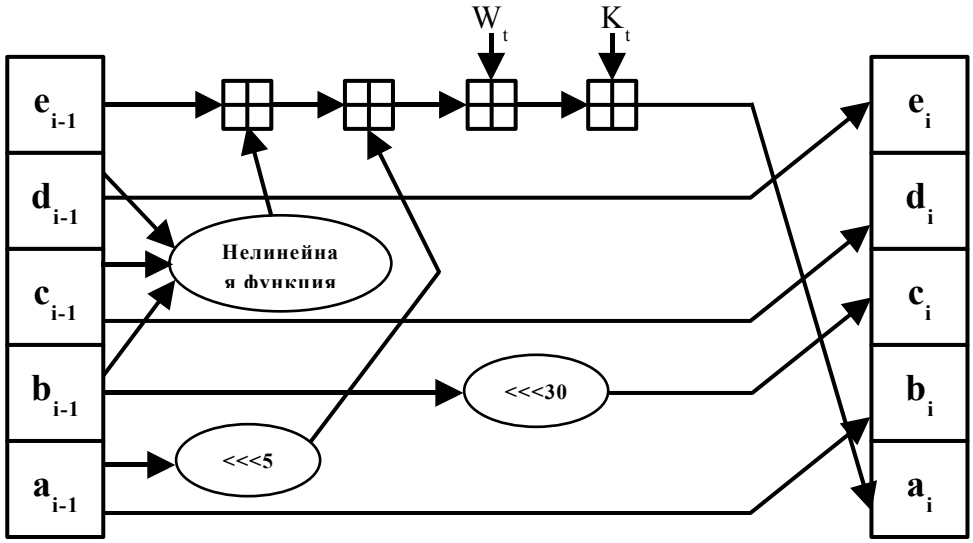

Рисунок 3.2

3.3.3 Алгоритмы ЭЦП

Алгоритм цифровой подписи RSA

Сначала необходимо вычислить пару ключей: секретный и открытый, используя алгоритм шифрования RSA.

Схему формирования и проверки ЭЦП RSA можно представить следующим образом (рисунок 3.3). [30]

Допустим, что отправителю необходимо подписать сообщение M. Тогда отправителем:

Вычисляется m=H(M), где m – целое число.

Вычисляется цифровая подпись

$S = m^{Ks} \bmod N$

Значения (M,S) отправляются получателю.

Получатель вычисляет хэш-функцию сообщения двумя способами:

1) $m' = S^{Ko} \bmod N$

2) $m = H(M)$

Если m'=m, то получатель признает подпись подлинной.

Недостатки алгоритма ЭЦП RSA:

–генерация ключей требует соблюдения большого количества условий;

–при вычислении ключей необходимо использовать большие числа, что требует значительных вычислительных затрат.

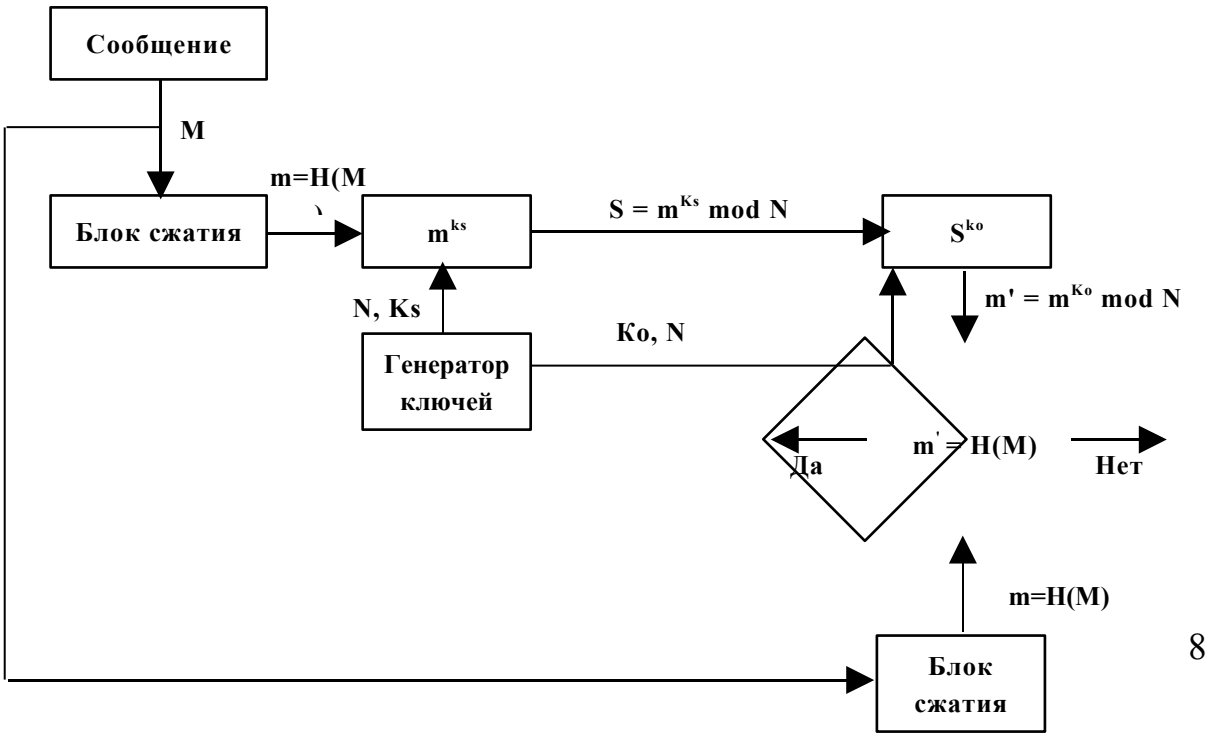

Рисунок 3.3

Алгоритм ЭЦП Эль Гамаля

В основе этого алгоритма лежит более сложная вычислительная задача, чем разложение числа на простые множители – задача дискретного логарифмирования.

Для генерации пары ключей выбирается простое большое число P и большое число G, G<P.

Отправитель сообщения вырабатывает случайное число x, 1<x<P и вычисляет

$Y= g^x \bmod P$

Число Y является открытым ключом, используемым для проверки подписи отправителя. Это число передается всем возможным получателям документов.

Число x является секретным ключом отправителя.

Чтобы подписать сообщение M, отправитель сначала хэширует его:

m=H(M), 1<m<H(M)

и генерирует случайное число k, 1<k<(P-1), причем (k, (P-1))=1.

Затем отправитель вычисляет

$a= G^k \bmod P$, а также с помощью секретного ключа x находит b из уравнения

$$m = (x \cdot a + k \cdot b) \bmod (P - 1)$$

Пара чисел a, b является цифровой подписью.

(M,a,b) передаются получателю.

Для проверки соответствия подписи документу получатель вначале вычисляет m=H(M).

Затем находит $A = Y^a \cdot a^b \bmod P$ и признает сообщение подлинным, если

$A=G^m \bmod P$.

3.4 Протоколы управления ключами

3.4.1 Хранение ключей

Под *функцией хранения ключей* понимают организацию их безопасного хранения, учета и удаления.

Ключ является самым привлекательным для злоумышленника объектом, открывающим ему путь к конфиденциальной информации. Поэтому вопросам безопасного хранения ключей следует уделять особое внимание.

Секретные ключи никогда не должны записываться в явном виде на носителе, который может быть считан или скопирован. Любая информация об используемых ключах должна быть защищена, в частности храниться в зашифрованном виде.

Необходимость в хранении и передаче ключей, зашифрованных с помощью других ключей, приводит к концепции *иерархии ключей.* В стандарте ISO 8532 (Banking-Key Management) подробно изложен метод главных/сеансовых ключей (master/session keys). Суть метода состоит в том, что вводится иерархия ключей: главный ключ (ГК), ключ шифрования ключей (КК), ключ шифрования данных (КД).

Иерархия ключей может быть:

–двухуровневой (КК/КД),

–трехуровневой (ГК/КК/КД),

Самым нижним уровнем являются рабочие или сеансовые КД, которые используются для шифрования данных, персональных идентификационных номеров (PIN) и аутентификации сообщений. Когда эти ключи надо зашифровать с целью защиты при передаче или хранении, используют ключи следующего уровня – ключи шифрования ключей. Ключи шифрования ключей никогда не должны использоваться как сеансовые (рабочие) КД, и наоборот.

Такое разделение функций необходимо для обеспечения максимальной безопасности. Фактически стандарт устанавливает, что различные типы рабочих ключей (например, для шифрования данных, для аутентификации и т.д.) должны всегда шифроваться с помощью различных версий ключей шифрования ключей.

В частности, ключи шифрования ключей, используемые для пересылки ключей между двумя узлами сети, известны также как *ключи обмена между узлами сети* (cross domain keys). Обычно в

канале используются два ключа для обмена между узлами сети, по одному в каждом направлении. Поэтому каждый узел сети будет иметь *ключ отправления* для обмена с узлами сети и *ключ получения* для каждого канала, поддерживаемого другим узлом сети

На верхнем уровне иерархии ключей располагается *главный ключ, мастер-ключ*. Этот ключ применяют для шифрования КК, когда требуется сохранить их на диске. Обычно в каждом компьютере используется только один мастер-ключ.

Мастер-ключ распространяется между участниками обмена неэлектронным способом - при личном контакте, чтобы исключить его перехват и/или компрометацию. Раскрытие противником значения мастер-ключа полностью уничтожает защиту компьютера.

Очень важным условием безопасности информации является периодическое обновление ключевой информации. При этом должны переназначаться как рабочие ключи, так и мастер-ключи. Вопрос обновления ключевой информации тесно связан с третьим элементом управления ключами – распределением ключей.

3.4.2 Распределение ключей

Распределение ключей - самый ответственный процесс в управлении ключами. К нему предъявляются следующие требования:

–оперативность и точность распределения;

–скрытность распределяемых ключей.

Распределение ключей между пользователями вычислительной сети реализуется двумя способами [30]:

–использованием одного или нескольких центров распределения ключей;

–прямым обменом сеансовыми ключами между пользователями сети.

Недостаток первого подхода состоит в том, что центру распределения ключей известно, кому и какие ключи распределены, и это позволяет читать все сообщения, передаваемые по сети. Возможные злоупотребления существенно влияют на защиту. При

втором подходе проблема состоит в том, чтобы надежно удостоверить подлинность субъектов сети.

Задача распределения ключей сводится к построению протокола распределения ключей, обеспечивающего:

–взаимное подтверждение подлинности участников сеанса;

–подтверждение достоверности сеанса механизмом запроса-ответа или отметки времени;

–использование минимального числа сообщений при обмене ключами;

–возможность исключения злоупотреблений со стороны центра распределения ключей (вплоть до отказа от него).

В основу решения задачи распределения ключей целесообразно положить принцип отделения процедуры подтверждения подлинности партнеров от процедуры собственно распределения ключей. Цель такого подхода состоит в создании метода, при котором после установления подлинности участники сами формируют сеансовый ключ без участия центра распределения ключей с тем, чтобы распределитель ключей не имел возможности выявить содержание сообщений.

Распределение ключей с участием центра распределения ключей. Протокол Kerberos

Характеристика протокола. Протокол Kerberos был разработан в Массачусетском технологическом институте в конце 1980-х гг. и получил свое название по имени трехголового пса Цербера из греческой мифологии, охранявшего вход в Аид. [19]

У протокола Kerberos, как и у легендарного пса из преисподней, тоже три «головы»: два абонента безопасности и доверенная третья сторона. В роли абонентов выступают два устройства (обычно это клиент и сервер), которые желают установить связь друг с другом. У каждого из них имеется свой долгосрочный уникальный пароль (например, тот, который пользователь предъявляет при входе в систему). Доверенная третья сторона — центр распределения ключей (Key Distribution Center, KDC) — служит посредником, снабжая обоих абонентов безопасности секретным ключом, который будет использоваться

ими сообща. KDC также знает пароли всех участников сеансов связи, которым он предлагает посреднические услуги. Эти пароли хранятся в зашифрованной базе данных.

Kerberos опирается на криптографию с секретными, или симметричными, ключами. Таким образом, обе участвующие стороны используют для шифрования и расшифровки направляемых друг другу сообщений общий секретный ключ.

Транзакция Kerberos состоит из манипулирования сеансовыми ключами и билетами. Сеансовый ключ — это секретный ключ, подготовленный специально для того, чтобы оба абонента безопасности могли совместно пользоваться одними и теми же идентификационными данными, и им была бы гарантирована целостность и конфиденциальность сообщений. Билет содержит важную информацию о транзакции и открывает запросившему его доступ к другому компьютеру. Клиент обязан получить сеансовый ключ и билет для каждого компьютера, к которому он хочет получить доступ.

В билете отмечается момент его создания и срок действия. Абоненты безопасности используют данные о времени создания билета в процессе аутентификации, поэтому их часы должны быть максимально точно синхронизированы друг с другом.

Срок действия билета отслеживается системой безопасности, для того чтобы ограничить масштабы возможного ущерба, который способен причинить злоумышленник, укравший каким-либо образом билет. Обычно билет действителен в течение рабочего дня, однако администратор может менять срок в сторону как уменьшения, так и увеличения, в зависимости от политики безопасности, осуществляемой в конкретной организации. При этом следует иметь в виду, что уменьшение срока действия может привести к увеличению объема трафика, так как абонентам безопасности придется чаще проходить аутентификацию.

Транзакция Kerberos. В среде Kerberos процесс аутентификации начинается с входа в систему. Пользователь А вводит на клиентской станции свое имя пользователя и пароль. Рабочая станция посылает эти данные А в центр распределения ключей. Там содержится главная база данных уникальных,

долгосрочных ключей всех участников сеансов связи данной зоны. KDC ищет главный ключ A (K_A), который он определяет на основании его пароля. Потом KDC создает сеансовый ключ (S_A), которым будет пользоваться совместно с A, и «билет на выдачу билета» (Ticket-Granting Ticket, TGT). Билет TGT включает вторую копию сеансового ключа (S_A), имя пользователя, введенное A, и срок действия. KDC шифрует этот билет своим собственным главным ключом (K_{KDC}), который известен только KDC – E_{KKDC}(TGT).

Затем KDC шифрует всю эту информацию (E_{KKDC}(TGT),S_A) с помощью главного ключа A (K_A) и возвращает полученный результат на ее рабочую станцию. Получив сообщение, которое выглядит как беспорядочный набор случайных битов, клиентская станция применяет к паролю A однонаправленную хэш-функцию, преобразующую пароль в главный ключ A (K_A). С его помощью станция расшифровывает полученный пакет и становится обладателем сеансового ключа (совместно с KDC) и билета TGT. Она может «забыть» главный ключ A — для общения с KDC ей будет достаточно сеансового ключа (S_A) и TGT.

Теперь предположим, что A пытается получить доступ к пользователю B. Вместо того чтобы установить с ним прямой контакт, станция A обращается в KDC. При этом она передает билет TGT, запрос на доступ к B и порцию данных под названием «аутентификатор» — специальную отметку времени. Аутентификатор шифруется с помощью сеансового ключа, который является общим для A и KDC (S_A) – E_{KKDC}(TGT), запрос на доступ к B, E_{SA}(аутентификатор).

KDC расшифровывает TGT, используя свой главный ключ (K_{KDC}). Как вы помните, в билете TGT содержится пользовательское имя A и копия общего сеансового ключа (S_A). KDC применяет этот сеансовый ключ (S_A) для расшифровки аутентификатора и поэтому может быть уверен, что данный запрос действительно поступил от A, потому что только A пользуется соответствующим общим сеансовым ключом (S_A).

Далее KDC создает два билета — один для A и один для B. Каждый билет содержит важную информацию, в том числе имя абонента безопасности, запросившего услугу, имя получателя

запроса, время создания билета и срок его действия. Кроме того, в оба билета вносится новый ключ (K_{AB}), который будет использоваться совместно A и B.

KDC шифрует билет B, используя его главный ключ (K_B). Затем билет B вкладывается в билет A, где наряду с прочими данными также содержится новый ключ (K_{AB}): KDC шифрует весь этот массив общим с A сеансовым ключом (S_A) и посылает результат A – E_{SA} (T_A, E_{KB}(T_B)).

Получив билет, A расшифровывает его с помощью сеансового ключа (S_A). Таким образом, у нее оказывается новый сеансовый ключ (K_{AB}) и билет B (который она не может прочитать). Затем A расшифровывает аутентификатор (отметку о времени) с помощью нового ключа (K_{AB}) и отправляет B аутентификатор вместе с его билетом – E_{KB}(T_B), E_{KAB}(аутентификатор A).

Получив все это, B сначала расшифровывает собственный билет, используя свой главный ключ (K_B). Это открывает доступ к новому сеансовому ключу (K_{AB}), который позволит расшифровать аутентификатор пользователя A.

Теперь и у A, и у B есть новый ключ (K_{AB}). B может не сомневаться, что A — именно тот пользователь, за которого он себя выдает, так как он зашифровал аутентификатор с помощью этого ключа (K_{AB}). Если у B возникнет необходимость ответить A, он воспользуется новым ключом (K_{AB}). A будет знать, что B — это именно B, и никто другой, потому что он мог получить новый ключ (KAB) только с помощью своего главного ключа (K_{AB}).

Аналогичным образом A может аутентифицировать себя при обращении к другим участникам. Для каждого запроса на установление связи KDC создаст уникальный сеансовый ключ, которым он пользуется совместно с инициатором запроса.

Версия Kerberos 5 заменила протокол Microsoft NT LAN Manager (NTLM) в роли основного метода сетевой аутентификации для Windows 2000. (NTLM остается в составе системы для совместимости с предыдущими версиями.) Microsoft отмечает ряд преимуществ Kerberos по сравнению с NTLM, включая более высокую эффективность аутентификации при доступе к серверам,

поддержку взаимной аутентификации, упрощенное управление доверительными отношениями и совместимость.

Прямой обмен ключами между пользователями. Алгоритм Диффи-Хеллмана

Алгоритм Диффи-Хеллмана был первым алгоритмом с открытыми ключами (предложен в 1976 г.). Его безопасность обусловлена трудностью вычисления дискретных логарифмов в конечном поле. [39]

Предположим, что два пользователя A и B хотят организовать защищенный коммуникационный канал.

1. Обе стороны заранее уславливаются о модуле N (N должен быть простым числом) и примитивном элементе g.

Эти два целых числа P и g могут не храниться в секрете. Как правило, эти значения являются общими для всех пользователей системы.

2. Затем пользователи A и B независимо друг от друга выбирают собственные секретные ключи K_A и K_B (случайные большие целые числа, которые хранятся пользователями A и B в секрете).

Далее пользователь A вычисляет открытый ключ

$Y_A=g^{kA} \bmod P$,

а пользователь B - открытый ключ

$Y_B=g^{kB} \bmod P$.

3. Затем стороны A и B обмениваются численными значениями открытых ключей Y_A и Y_B по незащищенному каналу.

4. Каждый из пользователей A и B вычисляет общий секретный ключ K соответственно:

$K=Y_B^{kA} \bmod P$

$K=Y_A^{kB} \bmod P$

Схема работы алгоритма Диффи-Хеллмана показана на рисунке 3.4.

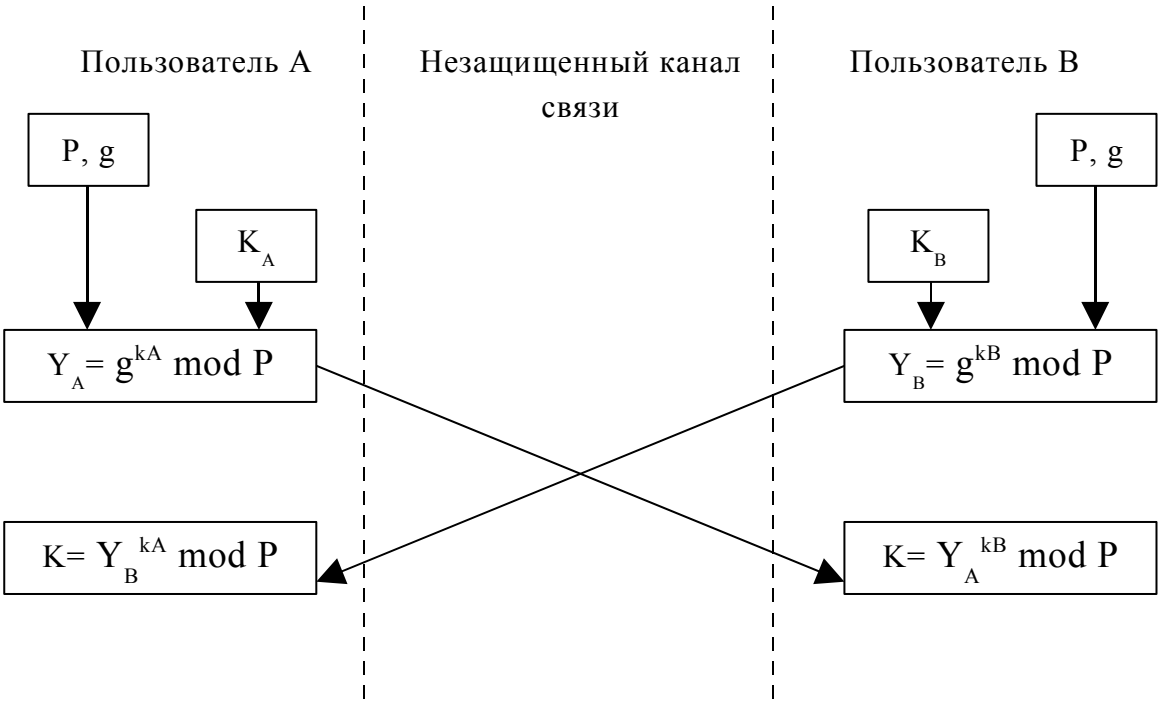

Рисунок 3.4

Выбор значений P и g может иметь существенное влияние на безопасность этой системы. Модуль P должен быть большим и простым числом. Число (P-1)/2 также должно быть простым числом. Число g желательно выбирать таким, чтобы оно было примитивным элементом множества ZP.

Алгоритм открытого распределения ключей Диффи-Хеллмана позволяет обойтись без защищенного канала для передачи ключей. Однако, работая с этим алгоритмом, необходимо иметь гарантию того, что пользователь А получил открытый ключ именно от пользователя В, и наоборот. Эта проблема решается с помощью электронной подписи, которой подписываются сообщения об открытом ключе.

3.5 Генерация ключей

3.5.1 Проблемы генерации случайных последовательностей

Прежде чем мы ознакомимся с некоторыми алгоритмами, используемыми для генерации ключевых последовательностей, охарактеризуем проблему.

Первый момент, на котором мы остановимся – длина симметричных и открытых ключей.

Безопасность симметричной криптосистемы основывается на надежности алгоритма и длины ключа. Предположим, что криптоалгоритм совершенен в том плане, что не остается другого способа вскрытия, кроме грубой силы. В этом случае криптоаналитику понадобится блок шифртекста и соответствующего открытого текста. Рассчитать сложность вскрытия нетрудно. Если используется 8-битовый ключ, следовательно? существует 2^8 или 256 возможных ключей. Если 56-битовый ключ – то 2^{56} возможных ключей. При условии проверки миллиона ключей в секунду, это займет 2285 лет. Таким образом, при допущении, что криптоалгоритм вскрыть можно только «грубой силой», длина ключа должна быть достаточно большой, чтобы сделать это вскрытие трудоемким.

Криптография с открытыми ключами использует однонаправленные алгоритмы, например, идею сложности разложения чисел на простые сомножители. Однако на самом деле математически не доказано, что указанная проблема трудно разрешима.

Асимметричные алгоритмы также подвержены вскрытию грубой силой, но по-разному. Вскрытие этих алгоритмов состоит не из перебора всех возможных ключей, а из попыток разложения большого числа на множители (или взятия дискретного логарифма в очень большом конечном поле). Если числа малы, то система никак не защищена, в противном случае – хорошо защищена.

Разложение на множители числа по-прежнему сложная задача, но со временем она упрощается, причем быстрее, чем это было предсказано математиками. В 1977 году Рон Ривест заявил, что разложение 125-разрядного числа потребует 40 квадриллионов лет. Однако в 1994 году было разложено 129-разрядное число. [42]

Управление ключами представляет собой самую трудную часть криптографии. Вскрыть безопасные криптографические протоколы непросто, но сохранить в секрете ключ намного

труднее. Можно не тратить временные и денежные ресурсы на вскрытие ключа, когда подкупить человека или запугать его окажется намного дешевле. Ничего не стоит вскрыть безопасность системы, где ключи хранятся ненадежно и т.п.

Большинство пользователей выбирают ключи, каким-либо образом связанные с их жизнью, работой: имена друзей, размер зарплаты и т.п.

Вскрытие «грубой силой» не перебирает все возможные ключи в числовом порядке, а пробует вначале очевидные ключи. Такое вскрытие называется вскрытие со словарем.

Вскрытие со словарем намного мощнее, если оно используется против файла ключей, а не против одного ключа. Если из 1000 пользователей каждый выберет собственный ключ (или пароль), то велика вероятность того, что, по крайней мере, один человек выберет ключ, имеющийся в словаре взломщика. Хорошими ключами являются строки случайных битов. Однако часто приходится запоминать ключ. Лучшим способом является использование целой фразы и преобразование этой фразы в ключ. Такие фразы называются ключевыми. Методика с названием «перемалывание ключа» преобразует легко запоминающиеся фразы в случайные ключи.

Если фраза достаточно длинная, то полученный ключ будет случаен. Но остается открытым вопрос, насколько длинной должна быть фраза.

Традиционно при генерировании последовательности якобы случайных чисел требуется, чтобы последовательность получаемых чисел была случайной в некотором вполне определенном статистическом смысле. Для проверки любой последовательности на случайность обычно служат два следующих критерия:

–однородность распределения: распределение чисел в последовательности должно быть однородным, т.е. частота появления в последовательности конкретного значения должна быть примерно одинаковой для всех значений;

–независимость: ни одно из значений последовательности не должно логически выводиться из других значений.

Имеются вполне четкие алгоритмы проверки того, что некоторая последовательность чисел соответствует заданному

распределению (например, однородному), но вот алгоритма, позволяющего "доказать" независимость, нет. Здесь обычно применяется ряд тестов, позволяющих продемонстрировать, что последовательность не является независимой. Общая стратегия состоит в применении таких тестов до тех пор, пока убеждение в независимости последовательности не станет достаточно правдоподобным.

В контексте нашей темы использование последовательности чисел, кажущихся статистически случайными, часто вытекает из криптографической структуры самого алгоритма шифрования. Например, одним из главных требований схемы RSA шифрования с открытым ключом является возможность генерирования простых чисел. Вообще-то совсем не просто определить, является ли данное достаточно большое число N простым или нет. Непосредственная проверка предполагает деление числа N на каждое целое число, меньшее N . Если N оказывается порядка, скажем, 10150, что не является нереальным для современных методов асимметричной криптографии, то такая проверка оказывается сегодня далеко за рамками аналитических знаний человека и вычислительных возможностей компьютера. Однако существует ряд эффективных алгоритмов проверки простоты числа с помощью использования последовательности случайно выбранных целых чисел в относительно простых вычислениях. Если такая последовательность достаточно длинна, простота тестируемого числа может быть определена почти наверняка. Такой подход, известный как рандомизация, служит для создания многих алгоритмов. По существу, если проблема слишком сложна или требует слишком много времени для точного решения, то для нахождения решения с любой требуемой долей уверенности применяется более простой и более быстрый способ, основанный на рандомизации.

В приложениях типа взаимной идентификации или генерирования сеансовых ключей требование статистической случайности последовательности чисел оказывается не настолько важным, насколько требование непредсказуемости элементов последовательности. В "истинно" случайной последовательности каждое число статистически независимо от других чисел последовательности и, таким образом, непредсказуемо. Однако

истинно случайные числа используются очень редко, чаще применяются последовательности чисел, которые выглядят случайными, но на самом деле генерируются с помощью некоторого алгоритма. В этом случае приходится заботиться о том, чтобы злоумышленник не имел возможности предсказать следующие элементы последовательности на основе предыдущих.

Нельзя сказать, что источники истинно случайных чисел являются очень распространенными устройствами. Потенциально такими источниками могут быть физические генераторы шумов, такие как импульсные детекторы ионизирующего излучения, газоразрядные лампы и конденсаторы с утечкой тока. Однако такие устройства могут найти весьма ограниченное применение в приложениях защиты информации.

Поэтому криптографические приложения обычно используют алгоритмические методы генерирования случайных чисел. Соответствующие алгоритмы являются детерминированными и поэтому порождают последовательности чисел, которые статистически не случайны. Однако если алгоритм достаточно хорош, порождаемые им последовательности чисел выдерживают многие разумные тесты на случайность. Такие числа часто называют псевдослучайными.

3.5.2 Криптографически генерируемые случайные числа

Один из лучших, с точки зрения криптографии, генераторов псевдослучайных чисел определяется стандартом X9.17. Предложенный алгоритм используется целым рядом приложений, среди которых приложения, обеспечивающие безопасность финансовых платежей и PGP [30, 39]. На рисунке 3.5 показана схема алгоритма, в котором для шифрования используется алгоритм DES, оперирующий 64-битовыми значениями.

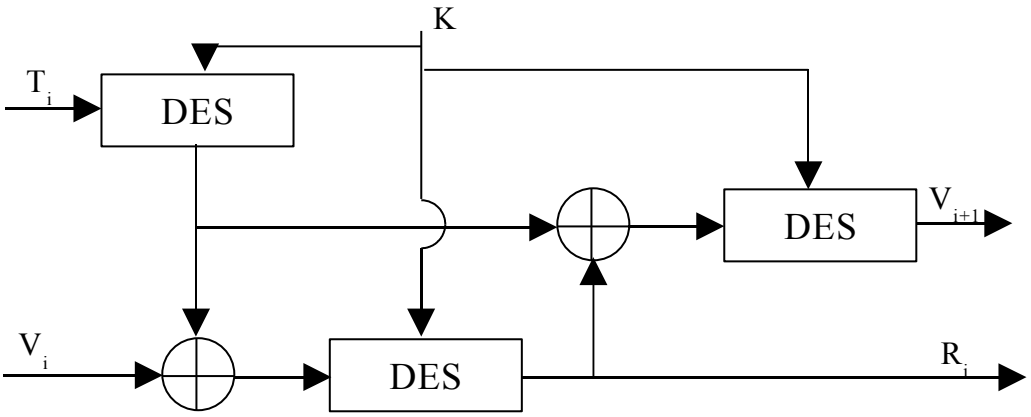

Рисунок 3.5

Ввод. На вход генератора подаются два псевдослучайных значения. Одно из них является 64-битовым представлением текущих даты и времени и меняется для каждого нового генерируемого числа. Другое представляет собой 64-битовое начальное значение, которое инициализируется к некоторому произвольному значению и обновляется в процессе вычислений.

Ключи. Генератор использует три модуля шифрования DES. Каждый из этих трех модулей использует один и тот же 56-битовый ключ, который должен храниться в секрете и использоваться только для генерирования псевдослучайных чисел.

Вывод. Выводимыми значениями являются 64-битовое псевдослучайной число и 64-битовое начальное значение.

Определим следующие величины:

T_i – значение даты времени в начале i-й стадии генерирования;

V_i – начальное значение для i-й стадии генерирования;

R_i – псевдслучайное число, получаемое в результате i-й стадии генерирования;

K – ключ DES, используемый на каждой стадии.

Тогда

$R_i = DES_K [V_i \text{ xor } DES_K [T_i]]$,

$V_{i+1} = DES_K [R_i \text{ xor } DES_K [T_i]]$

Криптографическая надёжность этого метода определяется несколькими факторами. Схема управляется двумя вводимыми псевдослучайными значениями: значением даты и времени и начальным значением, производимым генератором, но отличным от производимого генератором псевдослучайного значения. Таким образом, объём данных, которые должен анализировать противник, оказывается большим. Даже если ему станет известным псевдослучайное значение R_i, из R_i невозможно будет вывести V_{i+1}, поскольку для генерирования V_{i+1} используется дополнительная операция DES.

4 Политика, модели и стандарты безопасности вычислительных систем

4.1 Понятие политики безопасности

Интегральной характеристикой защищаемой вычислительной системы может являться политика безопасности – качественное выражение свойств защищенности в терминах, представляющих систему.

Под *политикой безопасности* мы будем понимать набор норм, правил и практических приемов, регулирующих порядок хранения и обработки ценной информации.

Политика безопасности включает:

–множество возможных операций над объектами;

–для каждой пары "субъект, объект" множество разрешенных операций, являющееся подмножеством всего множества возможных операций. [17, 27]

Посокольку защита вычислительной системы осуществляется не против абстрактного множества угроз, а против конкретных угроз, то корректнее вести речь об *адекватной политике безопасности*, которая обеспечивает достаточный уровень защищенности вычислительной системы. Чем выше уровень защищенности, тем большему количеству угроз может противодействовать вычислительная система.

Задача выбора и поддержания адекватной политики безопасности является одной из наиболее важных задач системного администратора. Если принятая в компьютеной системе политика безопасности неадекватна, это может привести не только к нарушению ее безопасности, но и к снижению надежности функционирования системы. С другой стороны, не всякая адекватная политика безопасности применима на практике. При формировании политики безопасности следует помнить о том, что чем выше уровень защищенности вычислительной системы, тем труднее в ней работать: [29]

–любая система, в которой предусмотрены функции защиты информации, требует от администраторов определенных усилий, направленных на поддержание адекватной политики безопасности. Чем больше в операционной системе защитных функций, тем больше времени и средств нужно тратить на поддержание защиты;

–чем больше функций защиты поддерживается в вычислительной системе, тем больше аппаратных ресурсов они потребляют, тем самым сокращая фонд ресурсов для пользовательских программ;

–поддержание слишком жесткой политики безопасности может негативно сказаться на надежности и эффективности функционирования компьютерной системы.

Таким образом, при определении адекватной политики безопасности не следует пытаться достигнуть максимально возможного уровня защищенности компьютерной системы. Следует стремиться к формированию такой политики безопасности, которая, с одной стороны, будет способна противостоять наиболее вероятным злоумышленным действиям и обеспечивать достаточный уровень защищенности, с другой стороны, не приведет к вышеописанным негативным эффектам.

Формирование адекватной политики безопасности зависит от целей использования вычислительной системы, ее конфигурации, программного обеспечения. Нельзя сказать, что существует единая адекватная политика безопасности на все случаи жизни. Несмотря на то, что большинство современных операционных систем и программных средств достаточно универсальны и могут применяться для решения самых разнообразных задач, угрозы безопасности могут быть совершенно различны. Например, для автоматизированных систем наиболее распространенной угрозой является угроза несанкционированного доступа, для web-сервера – DoS-атаки.

4.2 Этапы разработки политики безопасности

Определение и поддержание адекватной политики безопасности вычислительной системы в общем случае можно разделить на ряд этапов. В своих рассуждениях мы будем основываться на Руководящий документ «Руководство по разработке профилей защиты и заданий по безопасности», предложенный Гостехкомиссией России в 2003 г., и методике, представленной в [29]. Выделяют следующие этапы формирования политики безопасности.

1. Анализ угроз.

Администратор вычислительной системы среди возможных угроз выделяет наиболее опасные, защите от которых нужно уделять максимум сил и средств.

Идентификация активов.

Угрозы характеризуются следующими аспектами: источник угрозы; предполагаемый метод нападения; уязвимости, которые могут быть использованы при нападении, и активы (т.е. информация или ресурсы, подлежащие защите с помощью информационных технологий), подверженные нападению.

Различают владельцев активов – это субъекты, ответственные за сохранность активов в пределах вычислительной системы. При описании активов следует определить их владельцев.

Идентификация источников угроз.

На данном шаге необходимо рассмотреть следующие аспекты:

а) кто может быть по каким-либо причинам заинтересован в компрометации идентифицированных активов;

б) кто, с учетом занимаемой должности, имеет возможность компрометации идентифицированных активов;

в) каковы предполагаемые уровень технической компетентности, уровень возможностей нарушителя, доступные ресурсы для реализации угрозы и мотивация нарушителя.

Источники угроз, не связанные с деятельностью человека, а также угрозы, возникшие в результате неумышленных действий человека, также должны быть рассмотрены.

В данный список могут быть включены угрозы, непосредственно которым подсистема защиты не противостоит:

–физическое нападение;

–злоупотребление правами со стороны привилегированных пользователей;

–неправильное администрирование и функционирование вычислительной системы вследствие недостаточной квалификации или халатности администратора.

Идентификация методов нападения

Следующим шагом после идентификации активов, подлежащих защите, и источников угроз, является идентификация возможных методов нападения, приводящих к компрометации активов. Идентификация возможных методов нападения основывается на информации о среде безопасности ОО, например:

а) потенциальные уязвимости активов, которые могут быть использованы источниками угроз;

б) возможности нарушителей, имеющих доступ к среде безопасности ОО.

Потенциальные уязвимости активов организации могут быть идентифицированы путем соответствующего анализа уязвимостей среды безопасности ОО с учетом идентифицированных предположений о среде. Тем не менее, следует помнить, что такой анализ может не выявить все уязвимости, и поэтому нельзя недооценивать возможность наличия новых и необнаруженных угроз.

Влияние результатов анализа рисков на идентификацию угроз

Проведение анализа рисков целесообразно на этапе идентификации угроз. Методы анализа риска должны учитывать следующие аспекты:

а) вероятность и последствия компрометации активов с учетом:

– возможности реализации идентифицированных методов нападения;

– вероятности успешной реализации нападения;

– возможного ущерба (включая величину материального ущерба, явившегося результатом успешного нападения);

б) другие ограничения, например, правовые нормы и стоимость.

2. Формирование требований к политике безопасности (спецификация угроз).

Администратор определяет, какие средства и методы будут применяться для защиты от тех или иных угроз. Например, защиту от несанкционированного доступа к некоторому объекту операционной системы можно решать либо средствами разграничения доступа, либо криптографическими средствами, либо используя некоторую комбинацию этих средств, либо используя какие-то иные средства.

Администратор должен сделать подобный обзор для каждой угрозы безопасности, выбирая оптимальные средства защиты от каждой угрозы. Одновременно с этим администратор анализирует возможные побочные эффекты различных вариантов политики безопасности, оценивая, в какой мере в каждом варианте политики безопасности будут проявляться побочные негативные факторы. Как правило, администратору приходится идти на компромисс, смиряясь либо с недостаточной защищенностью операционной системы от отдельных угроз, либо с определенными трудностями пользователей при работе с системой. Результатом данного этапа является набор требований по защите от угроз.

Чтобы обеспечить четкую спецификацию угроз, необходимо учесть следующие аспекты (идентифицированные на предыдущем этапе):

а) источники угроз;

б) активы, подверженные нападению (например, конфиденциальные данные);

в) используемый метод нападения.

Далее приводятся примеры формулирования угроз:

Угроза 1. Нарушитель может получить неуполномоченный доступ к конфиденциальной информации либо ресурсам ограниченного использования, выдав себя за уполномоченного пользователя.

Угроза 2. Уполномоченный пользователь может получить доступ к конфиденциальной информации или ресурсам ограниченного использования, выдав себя за другого уполномоченного пользователя.

Если описание угрозы сопровождается подробным объяснением, то это будет способствовать более глубокому осознанию пользователем сущности угрозы. Так, в примерах угроз, изложенных выше, целесообразно дать пояснение, что активами, подверженными риску компрометации, являются информация и ресурсы, к которым пользователь (в том числе выдававший себя за конкретного уполномоченного пользователя) имеет доступ.

Описание угроз должно затрагивать только те потенциальные события, которые непосредственно могут привести к компрометации активов, требующих защиты. Поэтому не рекомендуется включать угрозы, например, следующего вида: «В системе могут существовать недостатки обеспечения безопасности». Такая формулировка угрозы не способствует пониманию пользователем проблем безопасности.

3. Формальное определение политики безопасности.

Администратор четко определяет, как конкретно должны выполняться требования, сформулированные на предыдущем этапе. Он решает, можно ли добиться выполнения этих требований только встроенными средствами вычислительной системы или потребуются дополнительные средства (аппаратное или программное обеспечение). Формулируются необходимые требования к конфигурации операционной системы, а также требования к конфигурации дополнительных пакетов защиты, если установка таких пакетов необходима. Кроме того, администратор должен предусмотреть порядок внесения необходимых изменений в политику безопасности в чрезвычайных ситуациях, например, при обнаружении факта несанкционированного входа в систему пользователя-злоумышленника. Результатом данного этапа является развернутый перечень настроек конфигурации вычислительной системы и дополнительных средств защиты.

4. Претворение в жизнь политики безопасности.

К началу этого этапа у администратора вычислительной системы имеется четкое представление о том, какой должна быть адекватная политика безопасности. Задачей данного этапа является приведение конфигурации вычислительной системы и

дополнительных средств защиты в соответствие с политикой безопасности, формально определенной на предыдущем этапе.

5. Поддержание и коррекция политики безопасности.

На данном этапе вычислительная система функционирует в соответствии с политикой безопасности, определенной на третьем этапе. В задачу администратора входит контроль за соблюдением политики безопасности и ее актуализацией по мере появления изменений в функционировании вычислительной системы.

4.3 Виды политик безопасности

Выделяют следующие виды политик безопасности. [13]

Дискреционная (Discretionary Access Control). Основой дискреционной (дискретной) политики безопасности является дискреционное управление доступом, которое определяется двумя свойствами:

–все субъекты и объекты должны быть идентифицированы;

–права доступа субъекта к объекту системы определяются на основании некоторого внешнего по отношению к системе правила.

К достоинствам дискреционной политики безопасности можно отнести относительную простоту реализации соответствующих механизмов, что объясняет достаточно широкое распространение этого типа политики безопасности.

К недостаткам относится статичность модели. Данная модель не учитывает динамику изменений состояния системы. Кроме этого, при использовании дискреционной политики безопасности возникает вопрос правил распространения прав доступа и анализа их влияния на безопасность вычислительной системы.

Мандатная (Mandatory Access Control). Основу мандатной политики безопасности составляет мандатное управление доступом, которое подразумевает, что:

–все субъекты и объекты системы должны быть однозначно идентифицированы;

–задан линейно упорядоченный набор меток секретности;

–каждому объекту системы присвоена метка секретности, определяющая ценность содержащейся в нем информации – уровень секретности в вычислительной системе;

–каждому субъекту системы присвоена метка секретности, определяющая уровень доверия к нему в вычислительной системе – максимальное значение метки секретности объектов, к которым субъект имеет доступ; метка секретности субъекта называется его уровнем доступа.

Основная цель мандатной политики безопасности – предотвращение утечки информации от объектов с высоким уровнем доступа к объектам с низким уровнем доступа. Для систем мандатного разграничения доступа задача проверки безопасности является алгоритмически разрешимой. Кроме того, по сравнению с компьютерными системами, построенными на основе дискреционной политики безопасности, для систем, реализующих мандатную политику, характерна более высокая степень надежности.

Политика безопасности информационных потоков. Основана на разделении всех возможных потоков между объектами системы на два непересекающихся множества: благоприятных и неблагоприятных. Цель реализации политики безопасности информационных потоков состоит в том, чтобы обеспечить невозможность возникновения в компьютерной системе неблагоприятных информационных потоков.

Данный вид политики безопасности в большинстве случаев используется в сочетании с политикой другого вида (дискреционной или мандатной) и на практике трудно реализуема.

Политика ролевого разграничения доступа. Базируется на дискреционной политике безопасности и является ее развитием. Политика предполагает выделение групп прав доступа (ролей) с учетом специфики их применения.

Задание ролей позволяет определить более четкие и понятные для пользователей компьютерной системы правила разграничения доступа, динамически изменяющиеся в процессе функционироания компьютерной системы.

Политика изолированной программной среды. Целью реализации данной политики безопасности является определение

порядка безопасного взаимодействия субъектов системы. Политика изолированной программной среды реализуется путем изоляции субъектов системы друг от друга и путем контроля порождения новых субъектов таким образом, чтобы в системе могли активизироваться только субъекты из предопределенного списка. При этом должна контролироваться целостность объектов системы, влияющих на функциональность активизируемых субъектов.

4.4 Модели безопасности

4.4.1 Основные виды моделей безопасности

Все математические модели безопасности компьютерных систем классифицируются следующим образом:

–модели систем дискреционного разграничения доступа;

–модели систем мандатного разграничения доступа;

–модели безопасности информационных потоков;

–модели ролевого разграничения доступа;

–субъектно-ориентированная модель изолированной программной среды.

4.4.2 Модель матрицы доступов

Модель матрицы доступов (ММД) используется для анализа системы защиты, реализующей дискреционную политику безопасности, и ее основного элемента – матрицы доступов. При этом система защиты представляется конечным автоматом, функционирующим согласно определенным правилам перехода. [13, 14]

Обозначим:

O – множество объектов системы;

S – множество субъектов системы ($S \subseteq O$);

R – множество прав доступа субъектов к объектам, например, права на чтение (read), права на запись (write), владения (own);

M – матрица доступа, строки которой соответствуют субъектам, а столбцы – объектам; $M[s,o] \subseteq R$ - права доступа субъекта s к объекту o.

Отдельный автомат, построенный согласно положениям ММД, будем называть системой. Функционирование системы рассматривается только с точки зрения изменений в матрице доступа. Возможные изменения определяются шестью примитивными операторами:

–"внести" право $r \in R$ в M[s,o] – добавление субъекту s права доступа r к объекту o. При этом в ячейку M[s,o] матрицы доступов добавляется элемент r.

–"удалить" право $r \in R$ из M[s,o] – удаление у субъекта s права доступа r к объекту o. При этом из ячейку M[s,o] матрицы доступов удаляется элемент r.

–"создать" субъекта s' – добавление в систему нового субъекта s'. При этом в матрицу доступов добавляется новая строка и новый столбец.

–"создать" объект o' – добавление в систему нового объекта. При этом в матрицу доступов добавляется новый столбец.

–"уничтожить" субъекта s' – удаление из системы субъекта s'. При этом из матрицы доступов удаляются соответствующие строка и столбец.

–"уничтожить" объект o' – удаление из системы объекта. При этом из матрицы доступов удаляется соответствующий столбец.

В результате выполнения примитивного оператора α осуществляется переход системы из состояния Q=(S, O, M) в новое состояние Q' = (S', O', M'). Данный переход обозначим $Q \mid_{\alpha} Q'$ (таблица 4.1).

Из примитивных операторов могут составляться команды. Каждая команда состоит из двух частей:

–условия, при которых выполняется команда (наличие условий необязательно);

–последовательности примитивных операторов.

Таким образом, запись команды имеет вид:

command C (x_1, …, x_k)
 if $r_1 \in M[x_{s1},x_{o1}]$ and … and $r_m \in M[x_{sm},x_{om}]$ then
  α_1;
  …
  α_n;
 end;

Таблица 4.1

Примитивный оператор	**Условия выполнения**	**Новое состояние системы**
"внести" право $r \in R$ в M[s,o]	$s \in S$ $o \in O$	S'=S O'=O M'[s,o]=M[s,o] ∪ {r}
"удалить" право $r \in R$ из M[s,o]	$s \in S$ $o \in O$	S'=S O'=O M'[s,o]=M[s,o] \ {r}
"создать" субъект s'	$s' \notin S$	S' = S ∪ {s'} O' = O ∪ {s'} (s,o) ∈ SxO ⇒ M'[s,o]=M[s,o]
"создать" объект o'	$o' \notin O$	S'=S O' = O ∪ {o'} (s,o) ∈ SxO ⇒ M'[s,o]=M[s,o]
"уничтожить" субъект s'	$s' \in S$	S' = S \ {s'} O' = O\ {s'} (s,o) ∈ S'xO' ⇒ M'[s,o]=M[s,o]
"уничтожить" объект o'	$o' \in O$ $o' \notin S$	S' = S O' = O\ {s'} (s,o) ∈ S'xO' ⇒ M'[s,o]=M[s,o]

Здесь r_1, …$r_m \in R$ – права доступа, α_1, …, α_n – последовательность примитивных операторов.

При выполнении команды C (x_1, …, x_k) система осуществляет переход от состояния Q в новое состояние Q'. Данный переход обозначим $Q \mid_{C\ (x1,\ \ldots,\ xk)} Q'$. При этом справедливо:

Q'=Q, если одно из условий команды C (x_1, …, x_k) не выполнено;

$Q'=Q_n$, если все условия команды выполнены и существуют состояния Q, …Q: $Q = Q_0 |_{\alpha 1} Q1 |_{\alpha 2} \ldots |_{\alpha n} Q_n$.

Пример. Команда создания субъектом s личного файла f.

command "создать файл" (s,f):
"создать" объект f;
"внести" право владения own в M[s,f];
"внести" право на чтение read в M[s,f];
"внести" право на запись write в M[s,f];
end;

Команда передачи субъекту s' права read на файл f его владельцем субъектом s.

command "передать право на чтение" (s, s', f)
if own ∈ M[s,f] then
"внести" право read в M[s',f];
end;

4.4.3 Классическая модель Белла-ЛаПадула

В настоящее время модель является основной для анализа систем, реализующих мандатное разграничение доступа. В классической модели Белла-ЛаПадула анализируются условия, при выполнении которых в компьютерной системе невозможно возникновение информационных потоков от объектов с большим уровнем конфиденциальности к объектам с меньшим уровнем конфиденциальности. [13]

Введем обозначения основных элементов модели:

S – множество субъектов;

O – множество объектов;

R – множество видов доступа (read, write, append, execute) и видов прав доступа;

$B = \{b \subseteq S \times O \times R\}$ – множество возможных множеств текущих доступов в системе;

$(L,\leq)$ – решетка уровней конфиденциальности. Например, L={U (unclussified), C (confidential), S (secret), TS (top secret)}, где U<C<S<TS;

$M_{|S|\times|O|}$ – матрица доступов, где $M[s,o]\subseteq R$ – права доступа субъекта s на объект o;

$(f_S,f_O,f_C)\in F=L^S\times L^O\times L^S$ – тройка функций (f_S,f_O,f_C), определяющих: $f_S:S\to L$ – уровень доступа субъекта; $f_O:O\to L$ – уровень конфиденциальности объекта; $f_C:S\to L$ – текущий уровень доступа субъекта, при этом для любого $s\in S$ справедливо неравенство $f_C(s)\leq f_S(s)$;

$V=B\times M\times F$ – множество состояний системы;

Q – множество запросов к системе;

D – множество ответов на запросы, например: D={yes, no, error};

$W\subseteq Q\times D\times V\times V$ – множество действий системы, где четверка $(q,d,v^*,v)\in W$ обозначает, что система на запрос q с ответом d перешла из состояния v в состояние v^*.

$N_0=\{0,1,2,\ldots\}$ – множество значений времени;

X – множество функций $x:N_0\to Q$, задающих все возможные последовательности запросов к системе;

Y – множество функций $y:N_0\to D$, задающих все возможные последовательности ответов системы на запросы;

Z – множество функций $z:N_0\to V$, задающих все возможные последовательности состояний системы.

Определение 1. $\Sigma(Q,D,W,z_0)\subseteq X\times Y\times Z$ называется системой, если для каждого $(x,y,z)\subseteq\Sigma(Q,D,W,z_0)$ выполняется условие для $t\in N_0$, $(x_t,y_t,z_{t+1},z_t)\in W$, где z_0 – начальное состояние системы. При этом каждый набор $(x,y,z)\subseteq\Sigma(Q,D,W,z_0)$ называется реализацией системы, а $(x_t,y_t,z_{t+1},z_t)\in W$ называется действием системы в момент времени $t\in N_0$.

В классической модели Белла-ЛаПадула рассматриваются следующие запросы, входящие в множество Q:

1) запросы изменения множества текущих доступов b;

2) запросы изменения функций f;

3) запросы изменения текущей структуры разрешения доступа в матрице M.

Конкретное решение по запросу включает в себя возможность производить следующие изменения в состоянии системы:

1) изменение текущего доступа:

–получить доступ (добавить тройку (субъект, объект, вид доступа) в текущее множество b);

–отменить доступ (удалить тройку из текущего множества b).

2) изменение функций уровня конфиденциальности:

–изменить уровень конфиденциальности объекта;

–изменить текущий уровень конфиденциальности субъекта;

3) изменения разрешения доступа

–дать разрешение на доступ (добавить право доступа в соответствующий элемент матрицы доступов M);

–отменить разрешение на доступ (удалить право доступа из соответствующего элемента матрицы доступов M).

Безопасность системы определяется с помощью свойств:

–ss – свойства простой безопасности;

–* – свойства звезда;

–ds – свойства дискреционной безопасности.

Определение 2. Доступ $(s,o,r)\in S\times O\times R$ обладает ss-свойством относительно $f=(f_S,f_O,f_C)\in F$, если выполняется одно из условий:

– $r\in\{execute,append\}$;

– $r\in\{read,write\}$ и $f_S(s)\geq f_O(o)$.

Определение 3. Состояние системы $(b,M,f)\in V$ обладает ss-свойством, если каждый элемент $(s,o,r)\in b$ обладает ss-свойством относительно f.

Определение 4. Доступ $(s,o,r)\in S\times O\times R$ обладает *-свойством относительно $f=(f_S,f_O,f_C)\in F$, если выполняется одно из условий:

– $r = execute$;

– $r = append$ и $f_O(o)\geq f_C(s)$;

– $r = read$ и $f_C(s)\geq f_O(o)$;

– $r - write$ и $f_C(s)\geq f_O(o)$.

Определение 5. Состояние системы $(b,M,f)\in V$ обладает *-свойством, если каждый элемент $(s,o,r)\in b$ обладает *-свойством относительно f.

Определение 6. Состояние системы $(b,M,f)\in V$ обладает *-свойством относительно подмножества $S'\subseteq S$, если каждый элемент $(s,o,r)\in b$, где $s\in S'$, обладает *-свойством относительно f.

Определение 7. Состояние системы $(b,M,f)\in V$ обладает ds-свойством, если для каждого элемента $(s,o,r)\in b$ выполняется условие $r\in M[s,o]$.

Определение 8. Состояние системы (b,M,f) называется безопасным, если оно обладает *-свойством относительно S', ss-свойством и ds-свойством.

Определение 9. Реализация системы $(x,y,z)\subseteq\Sigma(Q,D,W,z_0)$ обладает ss-свойством (*-свойством, ds-свойством), если в последовательности $(z_0,z_1,\ldots)$ каждое состояние обладает ss-свойством (*-свойством, ds-свойством).

Определние 10. Система $\Sigma(Q,D,W,z_0)$ обладает ss-свойством (*-свойством, ds-свойством), если каждая ее реализация обладает ss-свойством (*-свойством, ds-свойством).

Определение 11. Система $\Sigma(Q,D,W,z_0)$ называется безопасной, если она обладает ss-свойством, *-свойством, ds-свойством одновременно.

Замечания: из приведенных выше определений безопасности систем можно сделать несколько выводов:

1) из обладания системой ss-свойством следует, что в модели Белла-ЛаПадула выполняется запрет на чтение вверх, требуемый мандатной политикой безопасности. Кроме того, ss-свойство не допускает модификацию с использованием доступа write, если $f_S(s)<f_O(o)$. Таким образом, функция $f_S(s)$ определяет для субъекта s верхний уровень конфиденциальности объектов, к которым он потенциально может получить доступ read или write;

2) из обладания *-свойством следует, что если субъект s может понизить свой текущий доступ до $f_C(s)=f_O(o)$, то он может получить доступ write на объект o, но не доступ read на объекты o', чей уровень $f_O(o')>f_C(s)$. Хотя при этом, возможно, справедливо

неравенство $f_S(s) \geq f_O(o')$ и в каких-то других состояниях системы субъект s может получить доступ read на объект o'. Таким образом, *-свойство исключает появление в системе неблагоприятного информационного потока сверху вниз и соответствует требованиям мандатной политики безопасности.

4.4.4 Модель информационных потоков

Рассмотренная модель матрицы доступов может быть использована при построении и анализе детерминированных систем защиты, т.е. систем, которые не включают элементов, имеющих вероятностную природу. При исследовании систем, закономерности функционирования которых сложны или практически не поддаются описанию, целесообразно использовать элементы теории вероятностей. К числу таких систем можно отнести глобальные вычислительные сети, современные многозадачные, многопользовательские сетевые операционные системы.

Рассмотрим систему S, реализующую мандатное разграничение доступа. Предположим, что:

–в системе используются только два уровня секретности: высокий и низкий;

–все объекты системы делятся на две непересекающиеся группы: высокоуровневые объекты (H), имеющие право обрабатывать информацию высокого уровня секретности, и низкоуровневые (L).

Задача системы: не допустить возникновение информационных потоков от высокоуровневых объектов к низкоуровневым.

Пусть на множестве значений объектов системы задано вероятностное распределение, т.е. H и L являются случайными величинами. Для описания и анализа информационных потоков между ними воспользуемся понятиями теории вероятностей: независимостей и условного распределения.

С их помощью рассмотрим два подхода к определению безопасности информационных потоков, основанных на понятиях:

–информационной невыводимости;

–информационного невмешательства.

Определение 1. В системе присутствует информационный поток от высокоуровневых объектов H к низкоуровневым L, если некое возможное значение переменной в некотором состоянии низкоуровневого объекта невозможно одновременно с возможными значениями переменных состояний высокоуровневых объектов.

Определение 2. Система безопасна в смысле информационной невыводимости, если в ней отсутствуют информационные потоки вида, описанного в определении 1.

Более формально, это можно записать как $p(H)>0$, $p(L)>0$, то $p(H/L)>0$.

В традиционной модели информационного невмешательства требуется, чтобы низкоуровневая информация была независима от высокоуровневой, т.е. выполнялось равенство

$p(L/H)=p(L)$,

что при $p(H)>0$, $p(L)>0$ равносильно равенству

$p(H/L)=p(H)$

Однако, если ввести параметр времени, то ограничение, данное выше, слишком строгое.

Если

t – момент времени; t=0,1,2…,

L_t – состояние всех низкоуровневых объектов в момент времени t,

H_t – состояние всех высокоуровневых объектов в момент времени t,

то не требуется выполнение

$p(H_t/L_{t-1})=p(H_t)$

т.е. текущее значение низкоуровневых объектов может содержать информацию о последующих значениях высокоуровневых объектов.

Например, если низкоуровневым объектом является несекретный файл, обрабатываемый пользователем с низким уровнем доступа, а высокоуровневым объектом является журнал аудита, то значение файла и операции, совершаемые над ним пользователем на шаге t, могут отображаться на шаге t+1 в журнале аудита.

Казалось бы, необходимо потребовать, чтобы текущие значения низкоуровневых объектов не зависели бы от значений высокоуровневых объектов на предыдущих тактах работы системы, т.е. выполнялось

$t=1,2,\ldots$

$p(L_t/H_{t-1})=p(L_t)$,

что равносильно

$p(H_{t-1}/L_t)=p(H_{t-1})$

Здесь учитывается тот факт, что состояние системы влияет на последующие только через информацию, хранимую в объектах системы.

Следует отметить, что данный вариант определения соотношения L_t и H_{t-1} является слишком строгим, так как предполагает независимость этих потоков. Однако значения высокоуровневых объектов на текущем шаге часто оказывает влияние на значения низкоуровневых объектов на последующих шагах работы системы.

Например, для защиты низкоуровневых пользователей от сбоев системы все данные, записываемые ими в низкоуровневые файлы, предварительно копируются системой в высокоуровневый аудиторский файл. В результате, если в момент времени t значение низкоуровневого файла было x, то это означает, что в момент времени t-1 высокоуровневый файл аудита содержал значение x. Налицо зависимость значений L_t и H_{t-1}, но это не является угрозой безопасности.

Более целесообразным представляется подход, обеспечивающий невозможность накопления низкоуровневыми объектами новой информации о значениях высокоуровневых объектов. Более формально, необходимо потребовать, чтобы знание значений L_t и L_{t-1} не давало бы новой информации о H_{t-1}, т.е. должно выполняться равенство

$t=1,2,\ldots$

$p(L_t/H_{t-1},L_{t-1})=p(L_t/L_{t-1})$,

что равносильно

$p(H_{t-1}/L_t,L_{t-1})=p(H_{t-1}/L_{t-1})$

т.е. тем самым запрещается обратный информационный поток из L_t и H_{t-1}, но не запрещался поток из L_t в H_{t+1}. Кроме этого следует отметить, что, решая проблемы, обозначенные в рассмотренных выше примерах, последнее правило запрещает временные каналы утечки информации.

С учетом того, что состояние системы влияет на последующие состояния только через информацию, хранимую в объектах системы, дадим определение модели безопасности информационных потоков.

Определение 3. Система безопасна в смысле информационного невмешательства, если выполняется равенство

$p(L_t/H_s,L_s)=p(L_t/L_s)$,
где $s,t=0,1,2\ldots$ и $s<t$

Эта модель служит практическим примером подхода к построению системы защиты, которая разрешает корреляцию значений высокоуровневых и низкоуровневых объектов, но при этом остается безопасной.

4.5 Критерии защищенности компьютерных систем

4.5.1 Критерии оценки безопасности компьютерных систем Министерства обороны США

Наиболее известным стандартом безопасности компьютерных систем является документ под названием "Критерии безопасности компьютерных систем" (Trusted computer system evaluation criteria), разработанный Министерством обороны США в 1983 г. Этот документ более известен под неформальным названием "Оранжевая книга".

В «Оранжевой книге» предложены три категории требований безопасности: политика безопасности, аудит и корректность, в рамках которых сформулированы шесть базовых требований безопасности. Первые четыре требования направлены непосредственно на обеспечение безопасности информации, два последних – на качество средств защиты. [14, 40]

Политика безопасности.

Требование 1. *Политика безопасности.* Система должна поддерживать точно определенную политику безопасности. Возможность доступа субъектов к объектам должна определяться на основании их идентификации и набора правил управления доступом. Там, где это необходимо, должна использоваться политика мандатного управления доступом, позволяющая эффективно реализовать разграничение доступа к информации различного уровня конфиденциальности.

Требование 2. *Метки.* С объектами должны быть ассоциированы метки безопасности, используемые в качестве исходной информации для процедур контроля доступа. Для реализации мандатного управления доступом система должна обеспечивать возможность присваивать каждому объекту метку или набор атрибутов, определяющих степень конфиденциальности (гриф секретности) объекта и режимы доступа к этому объекту.

Подотчетность.

Требование 3. *Идентификация и аутентификация.* Все субъекты должны иметь уникальные идентификаторы. Контроль доступа должен осуществляться на основании результатов идентификации субъекта и объекта доступа, подтверждения подлинности их идентификаторов (аутентификации) и правил разграничения доступа. Данные, используемые для идентификации и аутентификации, должны быть защищены от несанкционированного доступа, модификации и уничтожения и должны быть ассоциированы со всеми активными компонентами компьютерной системы, функционирование которых критично с точки зрения безопасности.

Требование 4. *Регистрация и учет.* Для определения степени ответственности пользователей за действия в системе, все происходящие в ней события, имеющие значение с точки зрения безопасности, должны отслеживаться и регистрироваться в защищенном протоколе, т.е. должен существовать объект компьютерной системы, потоки от которого и к которому доступны только привилегированному пользователю. Система регистрации должна осуществлять анализ общего потока событий и выделять из него только те события, которые оказывают влияние на безопасность для сокращения объема протокола и повышения эффективности его анализа. Протокол событий должен быть надежно защищен от несанкционированного доступа, модификации и уничтожения.

Гарантии (корректность)

Требование 5. *Контроль корректности функционирования средств защиты.* Средства защиты должны содержать независимые аппаратные и/или программные компоненты, обеспечивающие работоспособность функций защиты. Это означает, что все средства защиты, обеспечивающие политику безопасности, управление атрибутами и метками безопасности, идентификацию и аутентификацию, регистрацию и учет, должны находиться под контролем средств, проверяющих корректность их функционирования. Основной принцип контроля корректности состоит в том, что средства контроля должны быть полностью независимы от средств защиты.

Требование 6. *Непрерывность защиты.* Все средства защиты должны быть защищены от несанкционированного вмешательства и/или отключения, причем эта защита должна быть постоянной и непрерывной в любом режиме функционирования системы защиты и компьютерной системы в целом. Данное требование распространяется на весь жизненный цикл компьютерной системы.

Согласно "Оранжевой книге" все защищенные компьютерные системы делятся на семь классов от D1 (минимальная защита, фактически отсутствие всякой защиты) до A1 (максимальная защита). Усиление требований осуществляется с постепенным смещением акцентов от положений, определяющих наличие в системе каких-то определенных механизмов защиты, к положениям, обеспечивающим высокий уровень гарантий того, что система функционирует в соответствии с требованиями политики безопасности (таблица 4.2).

Обозначения:

«-» - нет требований к данному классу;

«+»- новые или дополнительны требования;

«=» - совпадают с требованиями предыдущего класса.

Кратко охарактеризуем классы защищенности.

Группа D. Минимальная защита.

Никаких требований. К этому классу относятся те системы, которые не удовлетворяют требованиям высших классов.

Таблица 4.2

Базовые требования **«Оранжевой книги»**	Классы защищенности					
	C1	**C2**	**B1**	**B2**	**B3**	**A1**
1	**2**	**3**	**4**	**5**	**6**	**7**
Политика безопасности						
1. Дискреционная политика безопасности	+	+	+	=	=	=
2. Мандатная политика безопасности	-	-	+	+	=	=
3. Метки секретности	-	-	+	+	=	=
4. Целостность меток	-	-	+	=	=	=
5. Рабочие метки	-	-	-	+	=	=
6. Повторение меток	-	-	+	=	=	=
7. Освобождение ресурсов при повторном использовании объектов	-	+	=	+	=	=

Продолжение таблицы 4.2

1	2	3	4	5	6	7
8. Изолирование модулей	-	=	=	=	=	=
9. Пометка устройств ввода/вывода	-	-	+	=	=	=
10. Пометка читаемого вывода	-	-	+	=	=	-
Подотчетность						
11. Идентификация и аутентификация	+	+	=	=	=	=
12. Аудит	-	+	+	+	+	=
13. Защищенный канал (доверенный путь)	-	-	-	+	=	=
Гарантии						
14. Проектная спецификация и верификация	-	-	+	+	+	+
15. Системная архитектура	+	=	=	+	+	=
16. Целостность системы	+	=	=	=	=	=
17. Тестирование системы безопасности	+	+	+	+	+	=
18. Доверенное восстановление после сбоев	-	-	-	-	+	=
19. Управление конфигурацией системы	-	-	-	+	+	+
20. Доверенное дооснащение системы	-	-	-	+	+	=
21. Доверенное распространение	-	-	-	-	+	=
22. Анализ скрытых каналов	-	-	-	+	+	+
Документация						
23. Руководство пользователя	+	=	=	=	=	=
24. Руководство по конфигурированию системы защиты	+	+	+	+	+	=
25. Документация по тестированию	+	=	=	=	=	+
26. Проектная документация	+	=	+	+	=	+

Группа С. Дискреционная защита.

Класс С1. В операционной системе поддерживается дискреционное разграничение доступа. Пользователь, начинающий работать с системой, должен подтвердить свою подлинность (аутентифицироваться).

Класс С2. Выполняются все требования класса С1. Все субъекты и объекты системы имеют уникальные идентификаторы. Все действия всех субъектов доступа, не разрешенные явно, запрещены. События, потенциально опасные для поддержания защищенности операционной системы, регистрируются в специальном журнале (журнале аудита), работать с которым могут только привилегированные пользователи. Вся информация, удаляемая из оперативной памяти компьютера или с внешних

носителей информации, удаляется физически и не может быть в дальнейшем доступна ни одному субъекту доступа.

Группа B. Мандатное управление доступом.

Класс B1. Выполняются все требования класса C2. Поддерживается полномочное (мандатное) разграничение доступа к объектам системы. Поддерживается маркировка экспортируемой информации.

Класс B2. Выполняются все требования класса B1. Подсистема защиты системы реализует формально определенную и четко документированную модель безопасности. Осуществляется контроль скрытых каналов утечки информации. Интерфейс подсистемы защиты четко и формально определен, его архитектура и реализация полностью документированы. Выдвигаются более жесткие требования к идентификации, аутентификации и разграничению доступа.

Класс B3. Выполняются все требования предыдущего класса. Дополнительно должно быть обеспечено: поддержка администратора безопасности; механизм аудита должен быть обеспечен возможностью сигнализации о любых событиях, связанных с безопасностью; поддержка процедуры восстановления системы; дополнительные требования к документации системы.

Группа A. Верифицированная защита.

Класс A1. Критерии защиты данного класса не определяют дополнительные по сравнению с классом B3 требования к архитектуре или политике безопасности компьютерной системы.

Наиболее важные требования к классу A1 можно определить следующим образом:

–формальная модель политики безопасности должна быть четко определена и документирована, должно быть дано математическое доказательство того, что модель соответствует своим аксиомам и что их достаточно для поддержания заданной политики безопасности;

–должны быть использованы формальные технологии для демонстрации соответствия подсистемы заоиты модели политики безопасности, а также для анализа и выявления скрытых каналов.

Со времени опубликования «Оранжевой книги» прошло достаточно большое количество времени, в течение которого ряд положений данного документа устарело. Это связано, прежде всего, с интенсивным развитием компьютерных технологий, переходом от персональных ЭВМ типа IBM-360/370 к рабочим станциям, высокопроизводительным вычислительным комплексам и сетевой модели вычислений. Чтобы исключить некорректность некоторых положений «Оранжевой книги» было выпущено достаточно большое количество различных поправок, вспомогательных документов, что со временем привело к тому, что количество исправлений превысило объем исходного документа. Поэтому в 1995 г. Национальным центром компьютерной безопасности США был опубликован документ «Интерпретация критериев безопасности компьютерных систем», объединяющий все дополнения и разъяснения.

Следует отметить, что «Оранжевая книга» стала первой попыткой создать единый стандарт безопасности, рассчитанный на проектировщиков, разработчиков, потребителей и специалистов по сертификации систем безопасности компьютерных систем. В свое время этот документ явился значительным шагом в области безопасности информационных технологий и до сих пор используется в качестве руководящего документа при сертификации компьютерных систем обработки информации.

4.5.2 Руководящие документы Гостехкомиссии

Руководящие документы Гостехкомиссии. В 1992 г. Гостехкомиссия при Президенте Российской Федерации опубликовала пять руководящих документов, посвященных вопросам защиты вычислительных систем. [31-36]

1. Защита от несанкционированного доступа к информации. Термины и определения.

2. Концепция защиты средств вычислительной техники (СВТ) и автоматизированных систем (АС) от несанкционированного доступа (НСД) к информации.

3. Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации.

4. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации.

5. Временное положение по организации разработки, изготовления и эксплуатации программных и технических средств защиты информации от несанкционированного доступа в автоматизированных системах и средствах вычислительной техники.

Нас будут интересовать 2, 3 и 4 части.

Основные положения концепции защиты СВТ и АС от НСД к информации

Во втором документе излагается система взглядов и принципов, которые закладываются в основу проблемы защиты информации от несанкционированного доступа. Кратко охарактеризуем положения концепции защиты СВТ и АС от НСД.

Концепция предназначена для заказчиков, разработчиков и пользователей СВТ и АС, которые используются для обработки, хранения и передачи требующей защиты информации.

Концепция является методологической базой нормативно-технических и методических документов, направленных на решение следующих задач:

–выработка требований по защите СВТ и АС от НСД к информации;

–создание защищенных от НСД к информации СВТ и АС;

–сертификация защищенных СВТ и АС.

Концепция предусматривает существование двух относительно самостоятельных и, следовательно, имеющих отличие направлений в проблеме защиты информации от НСД: направление, связанное с СВТ, и направление, связанное с АС. Отличие двух направлений порождено тем, что СВТ

разрабатываются и поставляются на рынок лишь как элементы, из которых в дальнейшем строятся функционально ориентированные АС, и поэтому, не решая прикладных задач, СВТ не содержат пользовательской информации.

Помимо пользовательской информации при создании АС появляются такие отсутствующие при разработке СВТ характеристики АС, как полномочия пользователей, модель нарушителя, технология обработки информации.

В связи с этим, если понятия защищенность (защита) информации от НСД в АС и защищенность (защита) АС от НСД к информации эквивалентны, то в случае СВТ можно говорить лишь о защищенности (защите) СВТ от НСД к информации, для обработки, хранения и передачи которой оно предназначено.

При этом защищенность СВТ есть потенциальная защищенность, т.е. свойство предотвращать или существенно затруднять НСД к информации в дальнейшем при использовании СВТ в АС.

В концепции приводится понятие НСД, которое рассматривается как доступ к информации, нарушающий установленные правила разграничения доступа, с использованием штатных средств, предоставляемых СВТ или АС. Под штатными средствами понимается совокупность программного, микропрограммного и технического обеспечения СВТ или АС.

В концепции выделяются семь принципов защиты информации от НСД:

–защита СВТ и АС основывается на положениях и требованиях существующих законов, стандартов и нормативно-методических документов по защите от НСД к информации;

–защита СВТ обеспечивается комплексом программно-технических средств;

–защита АС обеспечивается комплексом программно-технических средств и поддерживающих их организационных мер;

–защита АС должна обеспечиваться на всех технологических этапах обработки информации и во всех режимах функционирования, в том числе при проведении ремонтных и регламентных работ;

–программно-технические средства защиты не должны существенно ухудшать основные функциональные характеристики АС (надежность, быстродействие, возможность изменения конфигурации АС);

–неотъемлемой частью работ по защите является оценка эффективности средств защиты, осуществляемая по методике, учитывающей всю совокупность технических характеристик оцениваемого объекта, включая технические решения и практическую реализацию средств защиты;

–защита АС должна предусматривать контроль эффективности средств защиты от НСД. Этот контроль может быть либо периодическим, либо инициироваться по мере необходимости пользователем АС или контролирующими органами.

К основным способам НСД относятся:

–непосредственное обращение к объектам доступа;

–создание программных и технических средств, выполняющих обращение к объектам доступа в обход средств защиты;

–модификация средств защиты, позволяющая осуществить НСД;

–внедрение в технические средства СВТ или АС программных или технических механизмов, нарушающих предполагаемую структуру и функции СВТ или АС и позволяющих осуществить НСД.

Источником всех угроз является нарушитель, модель которого также приведена в концепции. В качестве нарушителя рассматривается субъект, имеющий доступ к работе со штатными средствами АС и СВТ как части АС.

Нарушители классифицируются по уровню возможностей, предоставляемых им штатными средствами АС и СВТ. Выделяется четыре уровня этих возможностей.

Классификация является иерархической, т.е. каждый следующий уровень включает в себя функциональные возможности предыдущего.

Первый уровень определяет самый низкий уровень возможностей ведения диалога в АС - запуск задач (программ) из

фиксированного набора, реализующих заранее предусмотренные функции по обработке информации.

Второй уровень определяется возможностью создания и запуска собственных программ с новыми функциями по обработке информации.

Третий уровень определяется возможностью управления функционированием АС, т.е. воздействием на базовое программное обеспечение системы и на состав и конфигурацию ее оборудования.

Четвертый уровень определяется всем объемом возможностей лиц, осуществляющих проектирование, реализацию и ремонт технических средств АС, вплоть до включения в состав СВТ собственных технических средств с новыми функциями по обработке информации.

В своем уровне нарушитель является специалистом высшей квалификации, знает все об АС и, в частности, о системе и средствах ее защиты.

Помимо вышеперечисленного в концепции рассматриваются такие вопросы, как:

–основные направления обеспечения защиты от НСД;

–организация работ по защите от НСД.

Показатели защищенности СВТ от НСД

В четвертой части Руководящих документов рассматриваются показатели защищенности СВТ от НСД, в данном документе устанавливается классификация СВТ по уровню защищенности от НСД к информации на базе перечня показателей защищенности и совокупности описывающих их требований. Под СВТ понимается совокупность программных и технических элементов систем обработки данных, способных функционировать самостоятельно или в составе других систем. Показатели защищенности СВТ применяются к общесистемным программным средствам и операционным системам (с учетом архитектуры ЭВМ). Конкретные перечни показателей определяют классы защищенности СВТ. Уменьшение или изменение перечня показателей, соответствующего конкретному классу защищенности

СВТ, не допускается. Каждый показатель описывается совокупностью требований.

Устанавливается семь классов защищенности СВТ от НСД к информации. Самый низкий класс - седьмой, самый высокий - первый.

Классы подразделяются на четыре группы, отличающиеся качественным уровнем защиты:

–первая группа содержит только один седьмой класс;

–вторая группа характеризуется дискреционной защитой и содержит шестой и пятый классы;

–третья группа характеризуется мандатной защитой и содержит четвертый, третий и второй классы;

–четвертая группа характеризуется верифицированной защитой и содержит только первый класс.

Выбор класса защищенности СВТ для автоматизированных систем, создаваемых на базе защищенных СВТ, зависит от грифа секретности обрабатываемой в АС информации, условий эксплуатации и расположения объектов системы.

Показатели защищенности и требования к классам приведены в таблице 4.3.

Обозначения:

« - » - нет требований к данному классу;

« + » - новые или дополнительные требования,

« = » - требования совпадают с требованиями к СВТ предыдущего класса.

Классы защищенности АС

В третьей части Руководящих документов Гостехкомиссии дается классификация АС и требований по защите информации в АС различных классов.

Все автоматизированные системы разделяются на три группы, в каждой из которых вводится своя иерархия классов защиты. Всего вводится девять классов защиты.

Таблица 4.3

Наименование показателя	**Класс защищенности**					
	6	**5**	**4**	**3**	**2**	**1**
а	**б**	**в**	**г**	**д**	**е**	**ж**

Дискреционный принцип контроля доступа	+	+	+	=	+	=
Мандатный принцип контроля доступа	-	-	+	=	=	=
Очистка памяти	-	+	+	+	=	=
Изоляция модулей	-	-	+	=	+	=
Маркировка документов	-	-	+	=	=	=
Защита ввода и вывода на отчуждаемый физический носитель информации	-	-	+	=	=	=
Сопоставление пользователя с устройством	-	-	+	=	=	=
Идентификация и аутентификация	+	=	+	=	=	=
Гарантии проектирования	-	+	+	+	+	+
Регистрация	-	+	+	+	=	=
Взаимодействие пользователя с КСЗ	-	-	-	+	=	=
Надежное восстановление	-	-	-	+	=	=
Целостность КСЗ	-	+	+	+	=	=
Контроль модификации	-	-	-	-	+	=
Контроль дистрибуции	-	-	-	-	+	=
Гарантии архитектуры	-	-	-	-	-	+
Тестирование	+	+	+	+	+	+
Руководство для пользователя	+	=	=	=	=	=
Руководство по КСЗ	+	+	=	+	+	=
Тестовая документация	+	+	+	+	+	=
Конструкторская (проектная) документация	+	+	+	+	+	+

В таблице 4.4 приведены требования к подсистемам защиты для каждого класса защищенности.

Обозначения:

« - » - нет требований к данному классу;

« + » - есть требования к данному классу.

Таблица 4.4

Подсистемы и требования	Классы								
	3Б	**3А**	**2А**	**1Д**	**1Г**	**1В**	**1Б**	**1А**	**2Б**
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1. Подсистема управления доступом									

Продолжение таблицы 4.4

1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
1.1. Идентификация, проверка подлинности и контроль доступа субъектов:									
• в систему	+	+	+	+	+	+	+	+	+
• к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним	-	-	-	-	+	+	+	+	+

устройствам ЭВМ									
• к программам	-	-	-	-	+	+	+	+	+
• к томам, каталогам, файлам, записям, полям записей	-	-	-	-	+	+	+	+	+
1.2. Управление потоками информации	-	-	-	-	-	+	+	+	+
2. Подсистема регистрации и учета									
2.1. Регистрация и учет:				+	+	+	+	+	
• входа (выхода) субъектов доступа в (из) системы (узел сети)	+	+	+	-	+	+	+	+	+
• выдачи печатных (графических) выходных документов	+	-	-	-	+	+	+	+	+
• запуска (завершения) программ и процессов (заданий, задач)	-	-	-	-	+	+	+	+	+
• доступа программ субъектов доступа к защищаемым файлам, включая их создание и удаление, передачу по линиям и каналам связи	-	-	-	-	+	+	+	+	+
• доступа программ субъектов доступа к терминалам, ЭВМ, узлам сети ЭВМ, каналам связи, внешним устройствам ЭВМ, программам, томам, каталогам, файлам, записям, полям записей	-	-	-	-	-	+	+	+	+

Продолжение таблицы 4.4

1	2	3	4	5	6	7	8	9	10
• изменения полномочий субъектов доступа	-	-	-	-	-	+	+	+	-
• создаваемых защищаемых объектов доступа	-	-	-	+	+	+	+	+	+
2.2. Учет носителей информации	+	+	+	-	+	+	+	+	+
2.3. Очистка (обнуление, обезличивание) освобождаемых областей оперативной памяти ЭВМ и внешних накопителей	+	-	-	-	-	+	+	+	+
2.4. Сигнализация попыток нарушения защиты	-	-	-						-
3. Криптографическая подсистема				-	-	-	+	+	
3.1. Шифрование конфиденциальной информации	-	-	-	-	-	-	-	+	+
3.2. Шифрование информации, принадлежащей различным субъектам	-	-	-	-	-	-	+	+	-

доступа (группам субъектов) на разных ключах									
3.3. Использование аттестованных (сертифицированных) криптографических средств	-	-	-						+
4. Подсистема обеспечения целостности				+	+	+	+	+	
4.1. Обеспечение целостности программных средств и обрабатываемой информации	+	+	+	+	+	+	+	+	+
4.2. Физическая охрана средств вычислительной техники и носителей информации	+	+	+	-	-	+	+	+	+
4.3. Наличие администратора (службы) защиты информации в АС	-	-	-	+	+	+	+	+	+
4.4. Периодическое тестирование СЗИ НСД	+	+	+	+	+	+	+	+	+
4.5. Наличие средств восстановления СЗИ НСД	+	+	+	-	-	+	+	+	+
4.6. Использование сертифицированных средств защиты	+	-	-						+

Рассмотренные Руководящие документы Гостехкомиссии следует воспринимать как первую стадию формирования отечественных стандартов в области информационной безопасности. Очевидно, что на разработку этих документов оказали большое влияние критерии «Оранжевой книги». К недостаткам Руководящих документов можно отнести: ориентация на защиту от НСД и отсутствие требований к адекватности реализации политики безопасности. Вследствие этого средства защиты ориентируются на противодействие только внешним угрозам, а к структуре самой системы и ее функционированию не предъявляется четких требований.

5 Защита информации в операционных системах

5.1 Функции подсистемы защиты операционной системы

Опишем основные функции, выполняемые подсистемой защиты операционной системы. Эти функции следуют из требований класса C2 «Оранжевой книги» и практически релизуются во всех современных операционных системах. [29]

1. *Разграничение доступа.* Каждый пользователь системы имеет доступ только к тем объектам операционной системы, к которым ему предоставлен доступ в соответствии с текущей политикой безопасности.

2. *Идентификация и аутентификация.* Ни один пользователь не может начать работу с операционной системой, не идентифицировав себя и не предоставив системе аутентифицирующую информацию, подтверждающую, что пользователь действительно является тем, за кого он себя выдает.

3. *Аудит.* Операционная система регистрирует в специальном журнале события, потенциально опасные для поддержания безопасности системы. Записи об этих событиях могут просматривать в дальнейшем только администраторы операционной системы.

4. *Управление политикой безопасности.* Политика безопасности должна постоянно поддерживаться в адекватном состоянии, т.е. должна гибко реагировать на изменения условий функционирования операционной системы, требований к защите информации, хранимой и обрабатываемой в системе, и т.д. Управление политикой безопасности осуществляется администраторами системы с использованием соответствующих средств, встроенных в операционную систему.

Подсистема защиты практически никогда не представляет собой единый программный модуль. Как правило, каждая из перечисленных функций подсистемы защиты решается одним или несколькими программными модулями, что будет продемонстрировано на примере подсистемы защиты данных Windows 2000. Некоторые функции встраиваются непосредственно

в ядро операционной системы. Тем не менее, должен существовать четко определенный интерфейс между различными модулями подсистемы защиты, используемый при взаимодействии модулей для решения общих задач.

Рассмотрим типичные атаки на операционную систему.

Сканирование файловой системы. Данная атака является одной из наиболее простых в реализации, но в то же время одной из наиболее опасных. Суть атаки заключается в том, что злоумышленник просматривает файловую систему компьютера и пытается прочесть (или скопировать или удалить) все файлы подряд. В случае большого объема файловой системы велика вероятность ошибки администратора, из-за которой злоумышленник получит несанкционированный доступ к информации. Данная атака обычно осуществляется в автоматическом режиме с использованием специальных программных средств.

Кража или подбор ключевой информации. Возможны разные способы реализации данной атаки, начиная с простейшего случая, когда злоумышленник подсматривает пароль, набираемый пользователем, и, заканчивая, использованием специальных средств.

Если политика безопасности предъявляет слишком высокие требования к паролю пользователя (например, очень большое количество символов в пароле), возможны ситуации, когда пользователи станут записывать пароль на бумагу. В этом случае пароль рано или поздно становится добычей злоумышленника.

Наконец, если ключевая информация хранится пользователем на внешнем носителе (ключевая дискета, Touch Memory и т.д.), этот носитель может быть потерян и затем найден злоумышленником, а может быть просто украден.

Отказ в обслуживании. Жадными называются программы, преднамеренно захватывающие значительную часть ресурсов компьютера, в результате чего другие программы не могут выполняться или выполняются крайне медленно и неэффективно. Часто запуск жадной программы приводит к краху операционной системы.

5.2 Разграничение доступа

Объектом доступа (или просто объектом) будем называть любой элемент операционной системы, доступ к которому пользователей и других субъектов доступа может быть произвольно ограничен. Другими словами, возможность доступа к объектам операционной системы определяется не только архитектурой операционной системы, но и текущей политикой безопасности.

Методом доступа к объекту называется операция, определенная для некоторого объекта. Например, для файлов могут быть определены методы доступа «чтение», «запись», «добавление», «исполнение».

Субъектом доступа (или просто субъектом) будем называть любую сущность, способную инициировать выполнение операций над объектами (обращаться к объектам по некоторым методам доступа).

Владелец - субъект, которому принадлежит данный объект и который несет ответственность за конфиденциальность содержащейся в объекте информации, а также за целостность и доступность объекта. Обычно в операционных системах принято при создании объекта назначать владельцем субъекта, его создавшего, в дальнейшем владелец объекта может быть изменен с использованием соответствующего метода доступа к объекту. На владельца возлагается ответственность не только за целостность и конфиденциальность данных в объекте, но и за корректное ограничение прав доступа к данному объекту других субъектов.

Правом доступа к объекту будем называть право на выполнение доступа к объекту по некоторому методу или группе методов. В последнем случае право доступа дает субъекту возможность осуществлять доступ к объекту по любому методу из данной группы. Говорят, что субъект имеет некоторое право на доступ к объекту (или "субъект имеет некоторое право на объект"), если он имеет возможность осуществлять доступ к объекту по соответствующему методу или группе методов. Например, если

пользователь имеет возможность читать файл, говорят, что он имеет право на чтение этого файла.

Понятие метода доступа и понятие права доступа не идентичны. Например, в операционной системе UNIX право на запись в файл дает возможность субъекту обращаться к файлу как по методу "запись", так и по методу "добавление", при этом, поскольку право доступа "добавление" в UNIX отсутствует, невозможно разрешить субъекту операцию добавления, одновременно запретив операцию записи. [24]

Привилегия – это право субъекта на доступ по некоторому методу или группе методов ко всем объектам операционной системы, поддерживающим данный метод доступа. Например, если субъект операционной системы Windows 2000 имеет привилегию отлаживать программы, он имеет право доступа ко всем объектам типа "процесс" и "поток" по группе методов, используемых отладчиками при отладке программ.

Разграничением доступа субъектов к объектам является совокупность правил, определяющая для каждой тройки субъект-объект-метод, разрешен ли доступ данного субъекта к данному объекту по данному методу.

Суперпользователь –субъекта доступа, который имеет возможность игнорировать правила разграничения доступа к объектам.

Правила разграничения доступа, действующие в операционной системе, устанавливаются администраторами системы при определении текущей политики безопасности. Правила разграничения доступа должны удовлетворять следующим требованиям.

1. Правила разграничения доступа, принятые в операционной системе, должны соответствовать аналогичным правилам, принятым в организации, в которой установлена операционная система.

2. Правила разграничения доступа не должны допускать разрушающие воздействия субъектов доступа, не обладающих соответствующими привилегиями, на операционную систему, выражающиеся в несанкционированном изменении, удалении или

другом воздействии на объекты, жизненно важные для обеспечения нормального функционирования операционной системы.

3. Любой объект доступа должен иметь владельца. Невозмно присутствие в системе *объектов,* не имеющих владельца.

4. Присутствие *недоступных объектов* - объектов, к которым не может обратиться ни один субъект доступа ни по одному методу доступа, недопустимо.

5.1.2 Идентификация и аутентификация

В операционной системе, которая претендует на название защищенной, любой субъект доступа, перед началом работы с системой, должен пройти идентификацию, аутентификацию и авторизацию.

Идентификация субъекта доступа заключается в том, что субъект сообщает операционной системе информацию, которая идентифицировала бы его среди других пользователей системы. Наличие такой идентифицирующей информации говорит о том, что пользователь был зарегистрирован в системе и является легальным пользователем.

Аутентификация (подтверждение подлинности субъекта) субъекта доступа заключается в том, что субъект предоставляет операционной системе помимо идентифицирующей информации еще и аутентифицирующую информацию, подтверждающую, что он действительно является тем субъектом доступа, к которому относится идентифицирующая информация.

В современных операционных системах для идентифкации и аутентификации пользователей используют: логин (уникальное имя пользователя, которое он получает после регистрации в системе) и пароль (произвольная последовательность символов, которую задает сам пользователь и хранит ее в секрете).

Авторизация субъекта доступа происходит после успешной идентификации и аутентификации. При авторизации субъекта операционная система выполняет действия, необходимые для того, чтобы субъект мог начать работу в системе. Например, авторизация пользователя в операционной системе UNIX включает в себя порождение процесса, являющегося операционной

оболочкой, с которой в дальнейшем будет работать пользователь. В операционной системе Windows 2000 авторизация пользователя включает в себя создание маркера доступа пользователя, создание рабочего стола и запуск на нем от имени авторизуемого пользователя процесса Userinit, инициализирующего индивидуальную программную среду пользователя. Авторизация субъекта не относится напрямую к подсистеме защиты операционной системы. В процессе авторизации решаются чисто технические задачи, связанные с организацией начала работы в системе уже идентифицированного и аутентифицированного субъекта доступа. Несмотря на это, авторизация пользователя должна происходить только после успешного прохождения процедур идентификации и аутентификации.

Для обеспечения надежной защиты операционной системы пароли пользователей не должны храниться в операционной системе в открытом виде. Типовые схемы идентификации пользователей на основе паролей и способы хранения паролей в системе уже были рассмотрены выше.

Следует отметить некоторые параметры, которые поддерживаются современными операционными системами, и могут быть использованы администратором для уменьшения вероятности угроз кражи или подбора пароля.

1. Ограничение срока действия пароля. При применении данного метода каждый пользователь операционной системы обязан менять пароль через определенные интервалы времени. Максимальный срок действия пароля целесообразно ограничить 30 - 60 днями. Менее сильные ограничения не дают желаемого эффекта, а при использовании более сильных ограничений резко повышается вероятность того, что пользователь забудет свой пароль. После того как срок действия пароля истек, пользователь должен сменить свой пароль в течение некоторого времени после первого входа в систему по истечении этого срока. Если пользователь не сменил пароль за отведенное время, операционная система запрещает ему входить в систему до тех пор, пока это явно не разрешит администратор системы.

Срок действия пароля должен ограничиваться не только сверху, но и снизу. В противном случае пользователь, сменив

пароль, может немедленно вернуться к старому паролю, сменив пароль еще раз.

Также целесообразно проверять при каждой смене пароля уникальность нового пароля. Для этого операционная система должна хранить не только образ текущего пароля пользователя, но и образы последних 5-10 паролей, им применявшихся.

2. *Ограничения на содержание пароля.* Данный метод заключается в том, что пользователь может выбрать себе в качестве пароля не произвольную строку символов, а только строку, удовлетворяющую определенным условиям. Обычно используются следующие условия:

–длина пароля не должна быть меньше некоторого количества символов; в литературе по компьютерной безопасности и в документации по операционным системам обычно рекомендуется запрещать использование паролей короче 6-8 символов, но, с учетом быстрого прогресса вычислительной техники, в настоящее время целесообразно ограничивать длину паролей уже 10 - 14 символами;

–в пароль должно входить по крайней мере 5-7 различных символов;

–в пароль должны входить как строчные, так и заглавные буквы;

–пароль пользователя не должен совпадать с его именем;

–пароль не должен присутствовать в списке "плохих" паролей, хранимом в системе.

Как правило, администраторы операционной системы, могут варьировать эти ограничения как в пределах всей операционной системы, так и для отдельных пользователей. Например, если некоторое имя пользователя используется для гостевого входа, устанавливать ограничения на используемый пароль нецелесообразно.

3. *Блокировка терминала.* При использовании данного метода, если пользователь несколько раз подряд ошибся при вводе имени и пароля, терминал, с которого пользователь входит в систему, блокируется, и пользователь не может продолжать дальнейшие попытки входа в систему. Параметрами данного метода являются:

–максимально допустимое количество неудачных попыток входа в систему с одного терминала;

–интервал времени, после которого счетчик неудачных попыток входа обнуляется;

–длительность блокировки терминала (может быть сделана неограниченной - в этом случае блокировка терминала снимается только администратором системы).

4. Блокировка учетной записи пользователя. Этот метод отличается от предыдущего только тем, что блокируется не терминал, с которого пользователь входит в систему, а учетная запись пользователя.

5. *Пароль и отзыв.* При использовании этой схемы аутентификации при входе пользователя в систему операционная система выдает ему случайное число или строку, на которую пользователь должен дать правильный отзыв. Фактически паролем являются параметры алгоритма преобразования запроса операционной системы в корректный ответ пользователя. Эти параметры выбираются операционной системой случайным образом для каждого пользователя, что фактически сводит данную схему аутентификации к предыдущей.

5.1.3 Аудит

Необходимость включения в защищенную операционную систему функций аудита диктуется следующими обстоятельствами.

1. Подсистема защиты операционной системы, не обладая интеллектом, не способна отличить случайные ошибки пользователей от злонамеренных действий. Например, то, что пользователь в процессе входа в систему ввел неправильный пароль, может означать как случайную ошибку при вводе пароля, так и попытку подбора пароля. Но, если сообщение о подобном событии записано в журнал аудита, администратор, просматривая этот журнал, легко сможет установить, что же имело место на самом деле - ошибка легального пользователя или атака злоумышленника. Если пользователь ввел неправильный пароль всего один раз - это явная ошибка. Если же пользователь пытался

угадать собственный пароль 20 -30 раз - это явная попытка подбора пароля.

2. Администраторы операционной системы должны иметь возможность получать информацию не только о текущем состоянии системы, но и о том, как она функционировала в недавнем прошлом. Журнал аудита дает такую возможность, накапливая информацию о важных событиях, связанных с безопасностью системы.

3. Если администратор операционной системы обнаружил, что против системы проведена успешная атака, ему важно выяснить, когда была начата атака и каким образом она осуществлялась. При наличии в системе подсистемы аудита не исключено, что вся необходимая информация содержится в журнале аудита.

Большинство экспертов по компьютерной безопасности сходятся во мнении, что привилегия работать с подсистемой аудита не должна предоставляться администраторам операционной системы. Другими словами, множество администраторов и множество аудиторов не должны пересекаться. При этом создается ситуация, когда администратор не может выполнять несанкционированные действия без того, чтобы это тут же не стало известно аудиторам, что существенно повышает защищенность системы от несанкционированных действий администраторов.

Требования к аудиту

Подсистема аудита операционной системы должна удовлетворять следующим требованиям.

1. Только сама операционная система может добавлять записи в журнал аудита. Если предоставить это право какому-то физическому пользователю, этот пользователь получит возможность компрометировать других пользователей, добавляя в журнал аудита соответствующие записи.

2. Ни один субъект доступа, в том числе и сама операционная система, не имеет возможности редактировать или удалять отдельные записи в журнале аудита,

3. Только пользователи-аудиторы, обладающие соответствующей привилегией, могут просматривать журнал аудита.

4. Только пользователи-аудиторы могут очищать журнал аудита. После очистки журнала в него автоматически вносится запись о том, что журнал аудита был очищен, с указанием времени очистки журнала и имени пользователя, очистившего журнал. Операционная система должна поддерживать возможность сохранения журнала аудита перед очисткой в другом файле.

5. При переполнении журнала аудита операционная система аварийно завершает работу ("зависает"). После перезагрузки работать с системой могут только аудиторы. Операционная система переходит к обычному режиму работы только после очистки журнала аудита.

Для ограничения доступа пользователей к журналу аудита недостаточно использования обычных средств разграничения доступа. Дело в том, что в подавляющем большинстве операционных систем администраторы, используя свои привилегии, могут прочитать и изменить содержимое любого файла операционной системы. Поэтому для ограничения доступа к журналу аудита должны применяться специальные средства защиты.

Хотя операционные системы не требуют обязательного сохранения журнала аудита перед очисткой, это желательно делать. Наиболее удобно сохранять старые журналы аудита на WORM-дисках, поскольку эти диски допускают только однократную запись информации, и информация, записанная на такой диск, может быть стерта только при физическом повреждении диска.

Политика аудита

Политика аудита - это совокупность правил, определяющих то, какие события должны регистрироваться в журнале аудита. Для обеспечения надежной защиты операционной системы в журнале аудита должны обязательно регистрироваться следующие события:

–попытки входа/выхода пользователей из системы;

–попытки изменения списка пользователей;

–попытки изменения политики безопасности, в том числе и политики аудита.

При определении политики аудита не следует ограничиваться регистрацией событий из перечисленных классов.

Окончательный выбор того, какие события должны регистрироваться в журнале аудита, а какие не должны, возлагается на аудиторов. При этом политика аудита в значительной мере определяется спецификой информации, хранимой и обрабатываемой в операционной системе, и дать какие-либо рекомендации, не зная этой специфики, невозможно.

При выборе оптимальной политики аудита следует учитывать ожидаемую скорость заполнения журнала аудита. Если политика аудита предусматривает регистрацию слишком большого числа событий, это не только не повышает защищенность операционной системы, но, наоборот, снижает ее. Если новые записи добавляются в журнал аудита слишком часто, аудиторам будет трудно выделить в огромном объеме малоценной информации те события, которые на самом деле представляют угрозу безопасности системы. Кроме того, чем быстрее заполняется журнал аудита, тем чаще его нужно очищать, и тем больше вероятность временного выхода из строя операционной системы из-за переполнения журнала аудита.

Политику аудита не следует рассматривать как нечто неизменное, заданное раз и навсегда. Политика аудита должна оперативно реагировать на изменения в конфигурации операционной системы, в характере хранимой и обрабатываемой информации, и особенно на выявленные попытки атаки операционной системы. Если, например, с помощью аудита было обнаружено, что имела место попытка преодолеть защиту операционной системы, но основные принципы реализации этой атаки остались неясными, целесообразно изменить политику аудита таким образом, чтобы при дальнейших попытках осуществлять аналогичные атаки аудиторы получали более подробную информацию.

В целом политика аудита - это своего рода искусство, и выбор оптимальной политики в значительной мере определяется опытом и интуицией аудитора.

Интерактивное оповещение аудиторов

В некоторых операционных системах подсистема аудита помимо записи информации о зарегистрированных событиях в специальный журнал предусматривает возможность

интерактивного оповещения аудиторов об этих событиях. Когда аудитор начинает работу с операционной системой одного из компьютеров сети, операционные системы других компьютеров получают соответствующие сообщения, после чего при каждой регистрации события в журнале аудита одного из компьютеров сети копия информации об этом событии передается на терминал, с которым работает аудитор.

Класс событий, регистрируемых в журнале аудита, не обязательно должен совпадать с классом событий, информация о которых передается аудиторам интерактивно. Целесообразно так организовать интерактивное оповещение аудиторов, чтобы аудиторы получали оповещение только о наиболее важных событиях - в противном случае аудиторам будет трудно выделить в сплошном потоке сообщений по-настоящему полезную информацию.

Данная дополнительная функция подсистемы аудита позволяет аудиторам более оперативно реагировать на попытки преодоления злоумышленниками защиты операционной системы и тем самым повышает общую защищенность системы. С другой стороны, техническая реализация этой функции весьма трудоемка, из-за чего интерактивное оповещение аудиторов пока применяется редко.

5.2 Общая характеристика компонентов системы защиты Windows 2000

5.2.1 Структура подсистемы защиты информации Windows 20000

Ниже перечислены компоненты и базы данных, на основе которых реализуется защита в Windows 2000. [37]

Справочный монитор безопасности (Security Reference Monitor, SRM). Компонент исполнительной системы (\Winnt\System32\Ntoskrnl.exe), отвечающий за проверку прав

доступа к объектам, манипулирование привилегиями (правами пользователей) и генерацию сообщений аудита безопасности.

Подсистема локальной аутентификации (local security authentication subsystem, Lsass). Процесс пользовательского режима, выполняющий образ \Winnt\System32\Lsass.exe, который отвечает за политику безопасности в локальной системе, а также за аутентификацию пользователей и передачу сообщений аудита безопасности в Event Log. Основную часть этой функциональности реализует сервис локальной аутентификации Lsasrv (\Winnt\System32\Lsasrv.dll) –DLL-модуль, загружаемый Lsass.

База данных политики Lsass. База данных, содержащая параметры политики безопасности локальной системы. Она хранится в разделе реестра HKLM\SECURITY и включает следующую информацию: каким доменам доверена аутентификация попыток входа в систему, кто имеет права на доступ к системе и каким образом, кому предоставлены те или иные привилегии и какие виды аудита следует выполнять. База данных политики Lsass в том числе хранит регистрационные данные, применяемые для входа в домены и при вызове Win32 сервисов.

Диспетчер учетных записей безопасности (Security Accounts Manager, SAM). Набор подпрограмм, отвечающих за поддержку базы данных, которая содержит имена пользователей и группы, определенные на локальной машине. Служба SAM, реализованная как \Winnt\System32\Samsrv.dll, выполняется в процессе Lsass.

База данных SAM. База данных, содержащая информацию о локальных пользователях и группах вместе с их паролями и другими атрибутами. Эта база данных хранится в разделе реестра HKLM\SAM.

Active Directory. Служба каталогов, содержащая базу данных со сведениями об объектах в домене. *Домен* — это совокупность компьютеров, в том числе периферийного

оборудования, и сопоставленных с ними групп безопасности, которые управляются как единое целое. Сведения о паролях и привилегиях пользователей домена и их групп содержатся в Active Directory и реплицируются на компьютеры, выполняющие роль контроллеров домена. Сервер Active Directory, реализованный как \Winnt\System32\Ntdsa.dll. выполняется в процессе Lsass.

Пакеты аутентификации. DLL-модули, выполняемые в контексте процесса Lsass и реализующие политику аутентификации в Windows 2000. DLL аутентификации отвечают за проверку пароля и имени пользователя, а также за возврат Lsass (в случае успешной проверки) детальной информации о правах пользователя.

Процесс Logon (Winlogon). Процесс пользовательского режима (\Winnt\ System32\Winlogon.exe), отвечающий на поддержку SAS и управление сеансами интерактивного входа в систему. Например, при регистрации пользователя Winlogon создает оболочку — пользовательский интерфейс.

GINA (Graphical Identification and Authentication). DLL пользовательского режима, выполняемая в процессе Winlogon и применяемая для получения пароля и имени пользователя или PIN-кода смарт-карты. Стандартная GINA хранится в \Winnt\System32\Msgina.dll.

Служба сетевого входа в систему (Net Logon, Netlogon). Win32-сервис (\Winnt\System32\Netlogon.dll), выполняемый в Lsass и реагирующий на запросы сетевого входа от Microsoft LAN Manager 2 под управлением Windows NT (любых версий до Windows 2000). В этом случае аутентификация обрабатывается как при локальной регистрации — данные передаются Lsass для проверки. В Netlogon также встроена служба локатора, необходимая для поиска контроллеров домена.

И Kernel Security Device Driver (KSecDD). Библиотека функций режима ядра, реализующая интерфейсы LPC (local procedure call), которые используются другими компонентами защиты режима ядра — в том числе шифрующей файловой системой (Encrypting File System, EPS) — для взаимодействия с Lsass в пользовательском режиме. KsecDD содержится в \Winnt\System32\ Drivers\Ksecdd.sys.

На рисунке 5.1 показаны взаимосвязи между некоторыми из этих компонентов и базами данных, которыми они управляют.

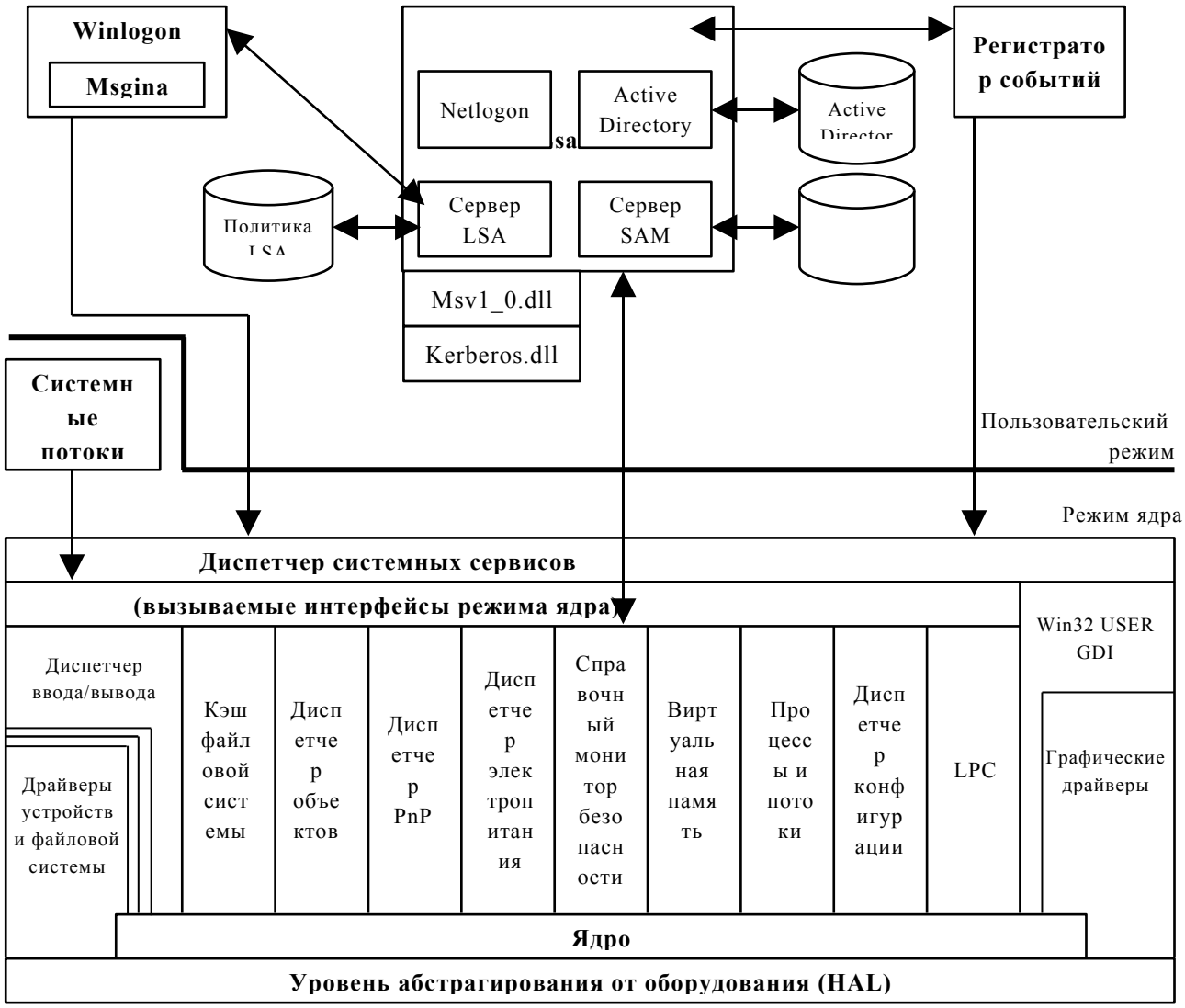

Рисунок 5.1

5.2.2 Объекты и субъекты доступа в Windows

В Windows 2000 (далее просто Windows), начиная с NT, все объекты операционной системы являются объектами доступа. [29, 37] Другими словами, доступ субъектов к любому объекту операционной системы может быть произвольно ограничен. Иерархия типов объектов имеет древовидную структуру, т.е. типы объектов могут иметь подтипы, эти подтипы, в свою очередь, могут иметь свои подтипы более низкого уровня, и т.д. Операции,

определенные над объектами некоторого типа, наследуются и объектами всех подтипов данного типа.

Рассмотрим стандартные типы объектов доступа.

Файловые объекты: файлы; дисковые директории - директории, расположенные на логических дисках; дисковые директории могут содержать файлы и другие дисковые директории; *устройства* - объекты, используемые для взаимодействия приложений с драйверами физических и логических устройств; *каналы,* или *трубы,* или *пайпы* (pipes), - объекты, используемые для организации взаимодействия процессов; *почтовые ящики* (mailslots) - объекты, используемые для асинхронной передачи сообщений между процессами.

Объектовые директории (object directories) - объекты, содержащие в себе другие объекты. В отличие от дисковых директорий объектовые директории могут содержать любые объекты. Объектовые директории являются временными объектами и хранятся только в оперативной памяти.

Ключи реестра (registry keys) - подмножества элементов конфигурации операционной системы.

Процессы - экземпляры программ, выполняющихся в данный момент на данном компьютере.

Потоки или *нити* (threads) - потоки машинных команд, последовательно выполняющиеся на процессоре. Один процесс может включать в себя несколько одновременно работающих параллельных потоков.

Диспетчер сервисов (service control manager) - объект Windows NT, используемый для управления сервисами.

Сервисы (services) - исполняемые модули, управляемые диспетчером сервисов. Сервисы используют программный интерфейс, несколько отличный от интерфейса, используемого обычными процессами и библиотеками. В программном интерфейсе Win32 к сервисам относятся как собственно сервисы, так и драйверы устройств. В "родном" программном интерфейсе ядра Windows NT управление драйверами осуществляется посредством управления соответствующими им устройствами. Таким образом, при выполнении запроса управления драйвером, инициированного из среды Win32, проверяется доступ к двум разным объектам - сервису и устройству, фактически представляющим собой единый объект - драйвер.

Объекты управления окнами: рабочие столы или *рабочие поля* (desktops) - совокупности взаимодействующих между собой окон; окна различных рабочих столов не могут быть видимы на экране компьютера одновременно; *оконные станции* (window stations) - совокупности рабочих столов; на разных оконных станциях могут одновременно работать разные пользователи.

Порты (ports) - объекты, используемые при передаче сообщений между процессами.

Секции разделяемой памяти (shared memory sections) или просто *секции* - области памяти, разделяемые между несколькими процессами.

Символические связи (symbolic links) - объекты, позволяющие создавать синонимы для имени объекта.

Маркеры доступа (access tokens) - объекты, содержащие информацию о работающих в системе пользователях и псевдопользователях.

Объекты синхронизации: события (events) - объекты, используемые при асинхронных обращениях к файловым системам и устройствам; *пары событий* (event pairs) - объекты, используемые при передаче сообщений от одного процесса к другому; *семафоры* (semaphores) - объекты, используемые для ограничения количества одновременных обращений разных потоков к одному объекту операционной системы; *мьютексы* (mutexes) - объекты, используемые для исключения одновременного доступа нескольких потоков к одному объекту операционной системы.

Файлы, дисковые директории и ключи реестра являются *постоянными* объектами и могут храниться на дисках компьютера. Остальные объекты являются временными и хранятся только в оперативной памяти.

Объекты, которые могут содержать другие объекты, а именно дисковые и объектовые директории, а также ключи реестра, называются *контейнерами.*

Windows позволяет прикладным и сервисным процессам создавать объекты доступа и других (нестандартных) типов. Перед созданием объекта нестандартного типа процесс должен зарегистрировать в системе данный тип объекта. Разграничение доступа субъектов к нестандартным объектам организуется в точности так же, как и к стандартным.

Перечислим типы *субъектов доступа*, которые поддерживает операционная система Windows (начиная с NT).

Пользователи - как обычные пользователи, так и псевдопользователи. К псевдопользователям относятся следующие субъекты доступа: SYSTEM -операционная система локального компьютера; этот псевдопользователь всегда входит в группу Administrators и имеет все привилегии; псевдопользователи с именами вида <имя_компьютера>$, где <имя_компьютера> - сетевое имя компьютера; эти псевдопользователи представляют операционные системы других компьютеров сети и используются при аутентификации рабочей станции на контроллере домена.

Группы пользователей. В Windows группы пользователей могут пересекаться, т.е. каждый пользователь Windows NT может входить в несколько групп. При этом для совместимости с программным интерфейсом POSIX, поддерживаемым Windows NT, среди групп, в которые входит пользователь, выделяется так называемая *первичная группа,* которая играет роль той единственной группы, в которую может входить пользователь в POSIX.

Специальные (временные) группы. В отличие от обычных групп членство пользователя в таких группах определяется операционной системой в зависимости от действий пользователя. Например, группа INTERACTIVE включает в себя пользователя, работающего с системой интерактивно, группа NETWORK - пользователей, работающих с системой через сеть, а группа DIALAP - пользователей, работающих с системой по модему. Специальная группа не может быть объявлена первичной группой пользователя.

Относительные субъекты. Эти субъекты имеют смысл только в применении к объекту, для которого определяются права доступа. Существуют следующие относительные субъекты:

CREATOR_OWNER - владелец объекта;

CREATOR_GROUP - первичная группа владельца объекта.

Относительные субъекты используются, если нужно описать права доступа пользователей к объектам по принципу "что кому принадлежит, то ему и доступно".

Следующие субъекты доступа Windows имеют предопределенные идентификаторы и присутствуют во всех экземплярах этой операционной системы: SYSTEM; Everyone - группа, в которую входят все пользователи и

псевдопользователи; INTERACTIVE; NETWORK; DIAL_UP; CREATOR_OWNER; CREATOR_GROUP.

Также существует несколько предопределенных идентификаторов, которые используются операционной системой внутренне или зарезервированы для последующих версий.

Идентификаторы остальных субъектов доступа уникальны.

5.2.3 Разграничение доступа в Windows

Модель защиты Windows 2000 требует, чтобы поток заранее — еще до открытия объекта — указывал, какие операции он собирается выполнять над этим объектом. Система проверяет тип доступа, запрошенный потоком, и, если такой доступ ему разрешен, он получает описатель, позволяющий ему (и другим потокам того же процесса) выполнять операции над объектом.

Одно из событий, заставляющее диспетчер объектов проверять права доступа, — открытие процессом существующего объекта по имени. При открытии объекта по имени диспетчер объектов ищет его в своем пространстве имен. Если этого объекта нет во вторичном пространстве имен (например, в пространстве имен реестра, принадлежащем диспетчеру конфигурации, или в пространстве имен файловой системы, принадлежащем драйверу файловой системы), диспетчер объектов вызывает внутреннюю функцию ObpCreateHandle. Как и подсказывает ее имя, он создает элемент в таблице описателей, который сопоставляется с объектом. Однако ObpCreateHandle создает описатель, обращаясь к функции исполнительной системы ExCreateHandle,только если другая функция диспетчера объектов, ObpIncrementHandleCount,сообщает, что поток имеет право на доступ к данному объекту. Реальную проверку прав доступа осуществляет другая функция диспетчера объектов, ObCbeckObjectAccess, которая возвращает результаты проверки функции ObpIncrementHandleCount.

ObpIncrementHandleCount передает ObCbeckObjectAccess удостоверения защиты потока, открывающего объект, типы запрошенного им доступа (чтение, запись, удаление и т. д.), а

также указатель на объект. ObCbeckObjectAccess сначала блокирует защиту объекта и контекст защиты потока. Блокировка защиты объекта предотвращает ее изменение другим потоком в процессе проверки прав доступа, а блокировка контекста защиты потока не дает другому потоку того же или другого процесса изменить идентификационные данные защиты первого потока при проверке его прав доступа. Далее ObCheckObjectAccess вызывает метод защиты объекта, чтобы получить параметры защиты объекта. Вызов метода защиты может привести к вызову функции из другого компонента исполнительной системы, но многие объекты исполнительной системы полагаются на стандартную поддержку управления защитой, предлагаемую системой.

Если компонент исполнительной системы, определяя объект, не собирается заменять стандартную политику безопасности, он помечает тип этих объектов. Всякий раз когда SRM вызывает метод защиты объекта, он сначала проверяет, использует ли объект стандартную защиту. Объект со стандартной защитой хранит информацию о защите в своем заголовке и предоставляет метод защиты с именем SeDefaultObjectMethod. Объект, не использующий стандартную защиту, должен сам поддерживать информацию о защите и предоставлять собственный метод защиты. Стандартную защиту используют такие объекты, как мьютексы, события и семафоры. Пример объекта с нестандартной защитой — файл. У диспетчера ввода-вывода, определяющего объекты типа «файл», имеется драйвер файловой системы, который управляет защитой своих файлов (или решает не реализовать ее). Таким образом, когда система запрашивает информацию о защите объекта «файл», представляющего файл в томе NTFS, она получает эту информацию от драйвера файловой системы NTFS, который в свою очередь получает ее от метода защиты объекта «файл», принадлежащего диспетчеру ввода-вывода. Заметьте, что при открытии файла ObCbeckObjectAccess не выполняется, так как объекты «файл» находятся во вторичных пространствах имен; система вызывает метод защиты объекта «файл», только если поток явно запрашивает или устанавливает параметры защиты файла (например, через Win32-функции SetFileSecurity или GetFileSecttrity).

Получив информацию о защите объекта, ObCheckObjectAccess вызывает SRM-функцию SeAccessCheck, на которую опирается вся модель защиты Windows 2000. Она принимает параметры защиты объекта, идентификационные данные защиты потока (в том виде, в каком они получены ObCheckObjectAccess) и тип доступа, запрашиваемый потоком. SeAccessCheck возвращает True или False в зависимости от того, предоставляет ли она потоку запрошенный тип доступа к объекту

Другое событие, заставляющее диспетчер объектов выполнять проверку прав доступа, — ссылка процесса на объект по существующему описателю. Подобные ссылки часто делаются косвенно, например, при манипуляциях с объектом через Win32 API с передачей его описателя. Допустим, поток, открывающий файл, запрашивает доступ для чтения из файла. Если у потока есть соответствующие права, определяемые его контекстом защиты и параметрами защиты файла, диспетчер объектов создает описатель данного файла в таблице описателей, которая принадлежит процессу — владельцу этого потока. Информация о предоставленном процессу типе доступа сопоставляется с описателем и сохраняется диспетчером объектов.

Впоследствии поток может попытаться что-то записать в этот файл через Win32-функцию WriteFile, передав в качестве параметра описатель файла. Системный сервис NtWriteFile, который WriteFile вызовет через Ntdll.dll, обратится к функции диспетчера объектов ObReferenceObjectByHandle ,чтобы получить указатель на объект «файл» по его описателю. ObReferenceObjectByHandle принимает запрошенный тип доступа как параметр. Найдя в таблице описателей элемент, соответствующий нужному описателю, ObReferenceObjectByHandle сравнит запрошенный тип доступа с тем, который был предоставлен при открытии файла. В данном случае ObReferenceObjectByHandle укажет, что операция записи должна завершиться неудачно, так как вызывающий поток, открывая файл, не получил право на его запись.

Функции защиты Windows 2000 также позволяют Win32-приложениям определять собственные закрытые объекты и вызывать SRM-сервисы для применения к этим объектам средств

защиты Windows 2000. Многие функции режима ядра, используемые диспетчером объектов и другими компонентами исполнительной системы для защиты своих объектов, экспортируются в виде Win32-функций пользовательского режима. Например, эквивалентом SeAccessCheck для пользовательского режима является AccessCbeck, Таким образом, Win32-приложения могут применять модель защиты Windows 2000 и интегрироваться с интерфейсами аутентификации и администрирования этой операционной системы.

Сущность модели защиты SRM отражает математическое выражение с тремя входными параметрами: идентификационными данными защиты потока, запрошенным типом доступа и информацией о защите объекта. Его результат — значения «да» или «нет», которые определяют, предоставит ли модель защиты запрошенный тип доступа.

5.2.4 Идентификаторы защиты

Для идентификации объектов, выполняющих в системе различные действия, Windows 2000 использует не имена (поскольку они не уникальны), а *идентификаторы защиты* (security identifiers, SID). SID имеются у пользователей, локальных и доменных групп, локальных компьютеров, доменов и членов доменов. SID представляет собой числовое значение переменной длины, формируемое из номера версии структуры SID, 48-битного кода агента идентификатора и переменного количества 32-битных кодов субагентов и/или *относительных идентификаторов* (relative identifiers, RID). Код агента идентификатора (identifier authority value) определяет агент, выдавший SID. Таким агентом обычно является локальная система или домен под управлением Windows 2000. Коды субагентов идентифицируют попечителей, уполномоченных агентом, который выдал SID, а RID — не более чем средство создания уникальных SID на основе общего базового SID (common-based SID). Поскольку длина SID довольно велика и Windows 2000 старается генерировать истинно случайные значения для каждого SID, вероятность появления двух одинаковых SID практически равна нулю.

В текстовой форме каждый SID начинается с префикса *S,* за которым следуют группы чисел, разделяемые дефисами, например:

S-1-5-21-1463437245-1224812800-863842198-1128

В этом SID номер версии равен 1, код агента идентификатора — 5 (Windows 2000), далее идут коды четырех субагентов и один RID в конце (1128). Этот SID относится к домену, так что локальный компьютер этого домена получит SID с тем же номером версии и кодом агента идентификатора; кроме того, в нем будет столько же кодов субагентов.

SID назначается компьютеру при установке Windows 2000 (программой Windows 2000 Setup). Далее Windows 2000 назначает SID локальным учетным записям на этом компьютере. SID каждой локальной учетной записи формируется на основе SID компьютера с добавлением RID. RID пользовательской учетной записи начинается с 1000 и увеличивается на 1 для каждого нового пользователя или группы. Аналогичным образом Windows 2000 выдает SID каждому только что созданному домену (работающему под управлением Windows 2000). Новые учетные записи домена получают SID, формируемые на основе SID домена с добавлением RID (который также начинается с 1000 и увеличивается на 1 для каждого нового пользователя или группы). RID с номером 1028 указывает на то, что его SID является 29-м, выданным доменом.

Многим предопределенным учетным записям и группам Windows 2000 выдает SID, состоящие из SID компьютера или домена и предопределенного RID Так, RID учетной записи администратора равен 500, а RID гостевой учетной записи — 501. Например, в основе SID учетной записи локального администратора лежит SID компьютера, к которому добавлен RID, равный 500:

S-1-5-21-13124455-12541255-61235125-500

Для групп Windows 2000 также определяет ряд встроенных локальных и доменных SID. Например, SID, представляющий любую учетную запись, называется Everyone и имеет вид S-1-1-0. Еще один пример — сетевая группа, т. е. группа, пользователи которой могут регистрироваться на данном компьютере из сети. SID сетевой группы имеет вид S-1-5-2.

5.2.5 Маркер доступа пользователя

В Windows каждый пользователь (в том числе и каждый псевдопользователь), работающий в системе, имеет свой маркер доступа. *Маркер доступа* (access token) - это объект специального вида, содержащий следующую информацию:

–идентификатор пользователя;

–идентификаторы групп и специальных групп, в которые входит пользователь;

–привилегии пользователя;

–идентификатор сеанса работы пользователя, к которому относится маркер доступа;

–атрибуты защиты, которые назначаются по умолчанию новым объектам операционной системы, созданным данным пользователем в текущем сеансе работы;

–имя и идентификатор подсистемы, выдавшей маркер доступа (Advapi, если пользователь вошел в систему локально, NtLogon, если пользователь вошел в систему по сети через SMB-сервер, и т.д.);

–некоторую служебную информацию.

Маркер доступа содержит всю информацию о пользователе, необходимую системе разграничения доступа для принятия решений о предоставлении пользователю доступа к тем или иным объектам операционной системы.

Каждому процессу Windows назначается так называемый *первичный маркер доступа* (primary access token). Обычно первичный маркер доступа - это маркер доступа пользователя, запустившего данный процесс. Субъект, обладающий соответствующей привилегией, может назначить процессу другой первичный маркер доступа. Отдельным потокам процесса могут назначаться свои маркеры доступа - *маркеры олицетворения* (impersonation access tokens). Механизм олицетворения обычно используется процессами-серверами. Когда процесс-сервер обслуживает запрос процесса-клиента, для выполнения запроса внутри процесса-сервера создается поток, которому назначается маркер доступа пользователя, инициировавшего запрос. В дальнейшем этот поток работает с правами того пользователя, от имени которого выполняется процесс-клиент.

Назначать маркер доступа процессу, а также создавать маркеры доступа могут только субъекты, обладающие соответствующими привилегиями.

Маркер доступа создается подсистемой аутентификации операционной системы в процессе авторизации пользователя, при этом псевдопользователь SYSTEM использует свою привилегию создавать маркеры доступа. После того как маркер доступа создан, информация о группах, в которые входит пользователь, и о привилегиях пользователя не может быть ни добавлена в маркер доступа, ни удалена из него. Однако группы и привилегии пользователя могут быть временно заблокированы (выключены). Если некоторая группа или привилегия заблокирована, подсистема разграничения доступа операционной системы при проверке прав доступа пользователя к объектам игнорирует наличие этой группы или привилегии в маркере доступа до тех пор, пока она не будет разблокирована.

В процессе регистрации в системе Winlogon создает начальный маркер, представляющий пользователя, который входит в систему, и сопоставляет его с процессом оболочки, применяемой для регистрации пользователя. Все программы, запускаемые пользователем, наследуют копию этого маркера. Длина маркеров варьируется из-за того, что учетные записи разных пользователей имеют неодинаковые наборы привилегий и сопоставлены с разными учетными записями групп. Но все маркеры содержат одну и ту же информацию, показанную на рисунке 5.2.

Механизмы защиты в Windows 2000 используют два элемента маркера, определяя набор действий, разрешенных потоку или процессу. Первый элемент состоит из SID учетной записи пользователя и полей SID групп. Используя SID-идентификаторы, SRM определяет, можно ли предоставить запрошенный тип доступа к защищаемому объекту, например к файлу в NTFS.

SID групп в маркере указывают, в какие группы входит учетная запись пользователя. При обработке клиентских запросов серверные приложения могут блокировать определенные группы для ограничения удостоверений защиты, сопоставленных с маркером. Блокирование группы дает почти тот же эффект, что и ее исключение из маркера.

Вторым элементом маркера, определяющим, что может делать поток или процесс, которому назначен данный маркер, является список привилегий – прав, составленных с маркером.

Источник маркера
Тип олицетворения

Идентификатор маркера
Идентификатор аутентификации
Идентификатор модификации
Время окончания действия
Основная группа по умолчанию
DACL по умолчанию
SID пользовательской учетной записи
SID группы 1
…
SID группы n
Ограниченный SID 1
…
Ограниченный SID n
Привилегия 1
…
Привилегия n

Рисунок 5.2

Поля основной группы маркера по умолчанию и списка управления дискреционным доступом (discretionary access-control list, DACL) представляют собой атрибуты защиты, применяемые Windows к объектам, которые создаются процессом или потоком с использованием маркера. Включая в маркеры информацию о защите, Windows упрощает процессам и потокам создание объектов со стандартными атрибутами защиты, так как в этом случае им не требуется запрашивать информацию о защите при создании каждого объекта.

Маркер может быть основным (primary token) (идентифицирует контекст защиты процесса) и олицетворяющим (impersonation token) (применяется для временного заимствования потоком другого контекста защиты — обычно другого пользователя). Маркеры олицетворения сообщают уровень олицетворения, определяющий, какой тип олицетворения активен в маркере.

Остальные поля маркера служат для информационных нужд. Поле источника маркера содержит сведения (в текстовой форме) о создателе маркера. Идентификатор маркера представляет собой локально уникальный идентификатор (locallyunique identifier, LUID), который SRM присваивает маркеру при его создании. Исполнительная система поддерживает свой LUID — счетчик, с помощью которого она назначает каждому маркеру уникальный числовой идентификатор.

Еще одна разновидность LUID — идентификатор аутентификации (authentication ID). Он назначается маркеру создателем. Как правило, Lsass является единственным создателем маркеров в системе и формирует LUID из LUID исполнительной системы. Lsass копирует идентификатор аутентификации для всех маркеров — потомков начального маркера. Используя этот идентификатор, программа может определить, принадлежит ли какой-то маркер тому же сеансу, что и остальные маркеры, анализируемые данной программой.

LUID исполнительной системы обновляет идентификатор модификации при каждом изменении характеристик маркера. Проверяя идентификатор модификации, программа может обнаруживать изменения в контексте защиты с момента его последнего использования.

5.2.6 Олицетворение

Олицетворение (impersonation) — мощное средство, часто используемое в модели защиты Windows 2000. Олицетворение также применяется в модели программирования «клиент-сервер». Например, серверное приложение может экспортировать ресурсы (файлы, принтеры или базы данных). Клиенты, которые хотят обратиться к этим ресурсам, посылают серверу запрос. Получив запрос, сервер должен убедиться, что у клиента есть разрешение на выполнение над ресурсом запрошенных операций. Так, если пользователь на удаленной машине пытается удалить файл с сетевого диска NTFS, сервер, экспортирующий этот сетевой ресурс, должен проверить, имеет ли пользователь право удалить данный файл. Казалось бы, что в таком случае сервер должен запросить учетную запись пользователя и SID-идентификаторы группы, а также просканировать атрибуты защиты файла. Но этот процесс труден для программирования, подвержен ошибкам и не позволяет обеспечить поддержку новых функций защиты. Поэтому Windows 2000 в таких ситуациях предоставляет серверу сервисы олицетворения.

Олицетворение позволяет серверу уведомить SRM о временном заимствовании профиля защиты клиента, запрашивающего ресурс, После этого сервер может обращаться к ресурсам от имени клиента, а SRM — проводить проверку его прав доступа. Обычно серверу доступен более широкий круг ресурсов, чем клиенту, и при олицетворении сервер может терять часть исходных

прав доступа. Также вероятно и обратное; при олицетворении сервер может получить дополнительные права.

За поддержку олицетворения в Windows 2000 отвечает несколько механизмов. Если сервер взаимодействует с клиентом через именованный канал, он может вызвать Win32-функцию *ImpersonateNamedPipeClient* и тем самым сообщить SRM о том, что ему нужно подменить собой пользователя на другом конце канала. Если сервер взаимодействует с клиентом через DDE (Dynamic Data Exchange) или RFC, то выдает аналогичный запрос на олицетворение через *DdeImpersonateClient* или *RpdmpersonateClient.* Поток может создать маркер олицетворения просто как копию маркера своего процесса, вызвав функцию *Imper-sonateSelf.* Для блокировки каких-то SID или привилегий поток может потом изменить полученный маркер олицетворения. Наконец, пакет SSPI (Security Support Provider Interface) может олицетворять своих клиентов через *ImpersonateSecurityContext.* SSPI реализует модель сетевой защиты вроде LAN Manager 2 или Kerberos.

После того как серверный поток завершает выполнение своей задачи, он возвращает себе прежний профиль защиты. Эти формы олицетворения удобны для выполнения определенных операций по запросу клиента. Их недостаток в том, что нельзя выполнять всю программу в контексте клиента. Кроме того, маркер олицетворения не дает доступа к сетевым файлам или принтерам, если только они не поддерживают null-сеансы. (Null-сеанс создается при анонимном входе.)

Если все приложение должно выполняться в контексте защиты клиента или получать доступ к сетевым ресурсам, клиент должен быть зарегистрирован в системе. Для этого предназначена Win32-функция *LogonUser,*которая принимает в качестве параметров имя учетной записи, пароль, имя домена или компьютера, тип регистрации (интерактивная, пакетная или сервисная) и провайдер регистрации (logon provider), а возвращает основной маркер. Серверный поток принимает маркер в виде маркера олицетворения, либо сервер запускает программу, основной маркер которой включает удостоверения клиента. С точки зрения защиты, процесс, создаваемый *LogonUser* для запуска программы в интерактивном сеансе регистрации, выглядит как программа, запущенная пользователем при интерактивном входе в систему.

Второй способ олицетворения контекста защиты клиента в Windows 2000 напоминает использование *LogonUser.* Он заключается в

дублировании маркера доступа клиента и применении его в качестве основного маркера, который передается *CreateProcessAsUser*. Недостаток способов, связанных с *LogonUser* и *CreateProcessAsUser,* в том, что серверу приходится получать имя и пароль по учетной записи пользователя. Если сервер передает эту информацию по сети, он должен надежно шифровать ее, чтобы избежать получения имени и пароля злоумышленником, перехватывающим сетевой трафик.

5.2.7 Дескриптор защиты

Атрибуты защиты объекта Windows 2000 описываются специальной структурой данных, называемой *дескриптором защиты* (security descriptor). Дескриптор защиты содержит следующую информацию:

–номер версии модели SRM, использованной для создания дескриптора;

–флаги: необязательные модификаторы, определяющие поведение или характеристики дескриптора;

–идентификатор владельца объекта;

–идентификатор первичной группы владельца объекта;

–список избирательного контроля доступа (discretionary access controllist, DACL) - список, полностью описывающий права различных субъектов на объект;

–системный список контроля доступа (system access control list, SACL) -используется при генерации сообщений аудита.

Если объект не имеет дескриптора защиты, при обращениях субъектов к нему права доступа не проверяются. В этом случае любой субъект имеет абсолютные права на данный объект.

Если объект хранится на диске или ином внешнем устройстве, дескриптор защиты хранится вместе с ним. При этом формат хранения объекта должен предоставлять такую возможность.

Файловые системы FAT и HPFS не поддерживают хранение на диске дескрипторов защиты для файлов. Только файлы и директории, расположенные на логических дисках с файловой системой NTFS, могут иметь дескрипторы защиты. Ключи реестра

могут иметь дескрипторы защиты независимо от файловой системы диска, на котором располагается реестр.

Список управления доступам (access-control list, ACL) состоит из заголовка и может содержать элементы (access-control entries, ACE). Существует два типа ACL DACL и SACL В DACL каждый ACE содержит SID и маску доступа (а также набор флагов), причем ACE могут быть четырех типов: «доступ разрешен» (access allowed), «доступ отклонен» (access denied), «разрешенный объект» (allowed-object) и «запрещенный объект» (denied-object). Первый тип ACE разрешает пользователю доступ к объекту, а второй — отказывает в предоставлении прав, указанных в маске доступа.

Разница между ACE типа «разрешенный объект» и «доступ разрешен», а также между ACE типа «запрещенный объект» и «доступ отклонен» заключается в том, что эти типы используются только в Active Directory. ACE этих типов имеют поле глобального уникального идентификатора (globally unique identifier, GUID), которое сообщает, что данный ACE применим только к определенным объектам или подобъектам (с GUID-идентификаторами). Кроме того, необязательный GUID указывает, что тип дочернего объекта унаследует ACE при его (объекта) создании в контейнере Active Directory, к которому применен ACE. (GUID — это гарантированно уникальный 128-битный идентификатор.)

ACE, используемые в DACL, также имеют набор флагов, контролирующих и определяющих характеристики ACE, связанные с наследованием. Некоторые пространства имен объектов содержат объекты-контейнеры и объекты-листы. Контейнер может включать другие контейнеры и листы, которые являются его дочерними объектами. Примеры контейнеров — каталоги в пространстве имен файловой системы и разделы в пространстве имен реестра. Отдельные флаги контролируют, как ACE применяется к дочерним объектам контейнера, сопоставленного с этим ACE.

SACL состоит из ACE двух типов: системного аудита (system audit ACE) и объекта системного аудита (system audit-object ACE). Эти ACE определяют, какие операции, выполняемые над объектами конкретными пользователями или группами, подлежат аудиту. Информация аудита хранится в системном журнале аудита. Аудиту могут подлежать как успешные, так и неудачные операции. Как и специфические для объектов ACE из DACL, ACE

объектов системного аудита содержат GUID, указывающий типы объектов или подобъектов, к которым применим данный ACE, и необязательный GUID, контролирующий передачу ACE дочерним объектам конкретных типов. При SACL, равном null, аудит объекта не ведется. Флаги наследования, применимые к DACL ACE, применимы к ACE системного аудита и объектов системного аудита.

Упрощенная схема объекта «файл» и его DACL представлена на рисунке 5.3.

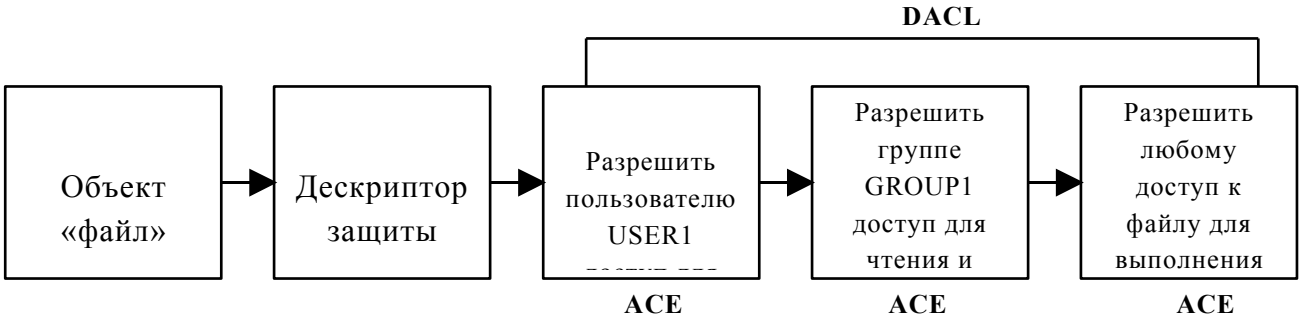

Рисунок 5.3 – Список управления избирательным доступом

Чтобы определить, какой DACL следует назначить новому объекту, система защиты использует первое применимое правило из следующего списка.

1. Если вызывающий поток явно предоставляет дескриптор защиты при создании объекта, то система защиты применяет его к объекту. Если у объекта есть имя и он находится в объекте-контейнере, система объединяет в DACL все наследуемые ACE (ACE, которые могут быть переданы от контейнера объекта), но только в том случае, если в дескрипторе защиты не установлен флаг, запрещающий наследование.

2. Если вызывающий поток не предоставляет дескриптор защиты и объекту присваивается имя, система защиты ищет этот дескриптор в контейнере, в котором хранится имя нового объекта. Некоторые ACE каталога объектов могут быть помечены как наследуемые. Это означает, что они должны применяться к новым объектам, создаваемым в данном каталоге. При наличии наследуемых ACE система защиты формирует из них ACL, назначаемый новому объекту.

3. Если дескриптор защиты не определен и объект не наследует какие-либо ACE, система защиты извлекает DACL по умолчанию из маркера доступа вызывающего потока и применяет его к новому объекту. В

некоторые подсистемы Windows 2000 (например, службы, LSA и SAM-объекты) «зашиты» свои DACL, назначаемые ими объектам при создании.

4. Если дескриптор защиты не определен и нет ни наследуемых ACE, ни DACL по умолчанию, система создает объект без DACL, что открывает полный доступ к нему любым пользователям и группам. Это правило идентично третьему, если маркер содержит нулевой DACL по умолчанию.

Правила, используемые системой при назначении SACL новому объекту, аналогичны правилам присвоения DACL за двумя исключениями. Первое заключается в том, что наследуемые ACE системного аудита не передаются объектам с дескрипторами защиты, помеченными флагом SE_SACL_PROTECTED (DACL точно так же защищается флагом SE_DACL_PROTECTED). Второе исключение: если ACE системного аудита не определены и наследуемого SACL нет, т о SACL вообще не присваивается объекту (в маркерах нет SACL по умолчанию).

Когда к контейнеру применяется новый дескриптор защиты, содержащий наследуемые ACE, система автоматически передает их в дескрипторы защиты дочерних объектов. В соответствии с порядком слияния наследуемых ACE с дескриптором защиты дочернего объекта любые ACE, явно примененные к ACL, размещаются до ACE, унаследованных объектом. Система использует следующие правила передачи наследуемых ACE:

–если дочерний объект без DACL наследует ACE, он получает DACL, содержащий лишь унаследованные ACE,

–если дочерний объект с пустым DACL наследует ACE, он также получает DACL, содержащий лишь унаследованные ACE;

–только для объектов в Active Directory: если наследуемый ACE удаляется из родительского объекта, все копии этого ACE автоматически удаляются из всех дочерних объектов,

–только для объектов в Active Directory; если из DACL дочернего объекта автоматически удалены все ACE, у дочернего объекта остается пустой DACL.

Для определения прав доступа к объекту используется два алгоритма:

–алгоритм, сравнивающий запрошенные права с максимально возможными для данного объекта и экспортируемый в пользовательский режим в виде Win32-функции *GetEffectiveRightsFromAcl;*

–алгоритм, проверяющий наличие конкретных прав доступа и активизируемый через Win32 -функцию *AccessCheck* или *AccessCheckByType,*

Первый алгоритм проверяет элементы DACL следующим образом.

В отсутствие DACL (DACL null) объект является незащищенным, и система защиты предоставляет к нему полный доступ.

Если у вызывающего потока имеется привилегия на захват объекта во владение (take-ownership privilege), система защиты предоставляет владельцу право на доступ для записи (write-owner access) до анализа DACL.

Если вызывающий поток является владельцем объекта, ему предоставляются права управления чтением (read-control) и доступа к DACL для записи (write-DACL access).

Из маски предоставленных прав доступа удаляется маска доступа каждого ACE типа «доступ отклонен», SID которого совпадает с SID маркера доступа вызывающего потока.

К маске предоставленных прав доступа добавляется маска доступа каждого ACE типа «доступ разрешен», SID которого совпадает с SID маркера доступа вызывающего потока (исключение составляют права доступа, в предоставлении которых уже отказано).

После анализа всех элементов DACL рассчитанная маска предоставленных прав доступа возвращается вызывающему потоку как максимальные права доступа. Эта маска отражает полный набор типов доступа, которые этот поток сможет успешно запрашивать при открытии данного объекта.

Все сказанное применимо лишь к той разновидности алгоритма, которая работает в режиме ядра. Его Win-32-версия, реализованная функцией *GetEffectiveRightsFromAcl,*отличается отсутствием шага 2, а также тем, что вместо маркера доступа она рассматривает SID единственного пользователя или группы.

Второй алгоритм проверяет, можно ли удовлетворить конкретный запрос на доступ исходя из маркера доступа вызывающего потока. У каждой Win32-функции открытия защищенных объектов есть параметр, указывающий желательную маску доступа — последний элемент выражения, описывающего защиту объектов. Чтобы определить, имеет ли вызывающий поток право на доступ к защищенному объекту, выполняются следующие операции.

В отсутствие DACL (DACL = null) объект является незащищенным, и система защиты предоставляет к нему запрошенный тип доступа.

Если у вызывающего потока имеется привилегия на захват объекта во владение, система защиты предоставляет владельцу право на доступ для записи, а затем анализирует DACL. Однако, если такой поток запросил только доступ владельца для записи, система защиты предоставляет этот тип доступа и не просматривает DACL.

Если вызывающий поток является владельцем объекта, ему предоставляются права управления чтением и доступа к DACL для записи. Если вызывающий поток запросил только эти права, система защиты предоставляет их без просмотра DACL.

Просматриваются все ACE в DACL — от первого к последнему. Обработка ACE выполняется при одном из следующих условий:

а) SID в ACE совпадает с незаблокированным SID (SID могут быть незаблокированными и заблокированными) в маркере доступа вызывающего потока независимо от того, какой это SID — основной или группы;

б) SID в ACE типа «доступ разрешен» совпадает с SID в маркере доступа вызывающего потока, и этот SID не имеет атрибута проверки только на запрет;

в) идет уже второй проход поиска в дескрипторе ограниченных SID, и SID в ACE совпадает с ограниченным SID в маркере доступа вызывающего потока.

В случае ACE типа «доступ разрешен» предоставляются запрошенные права из маски доступа ACE; проверка считается успешной, если предоставляются все запрошенные права. Доступ к объекту не предоставляется в случае ACE типа «доступ отклонен» и отказа в предоставлении какого-либо из запрошенных прав.

Если достигнут конец DACL и некоторые из запрошенных прав доступа еще не предоставлены, доступ к объекту запрещается.

Если все права доступа предоставлены, но в маркере доступа вызывающего потока имеется хотя бы один ограниченный SID, то система повторно сканирует DACL в поисках ACE, маски доступа которых соответствуют набору запрошенных прав доступа. При этом также идет поиск ACE, SID которых совпадает с любым из ограниченных SID вызывающего потока. Поток получает доступ к объекту, если запрошенные права доступа предоставлялись после каждого прохода по DACL.

Поведение обоих алгоритмов проверки прав доступа зависит от относительного расположения разрешающих и запрещающих ACE. Возьмем

для примера объект с двумя ACE, первый из которых указывает, что определенному пользователю разрешен полный доступ к объекту, а второй отказывает в доступе. Если разрешающий ACE предшествует запрещающему, пользователь получит полный доступ к объекту. При другом порядке этих ACE пользователь вообще не получит доступа к объекту.

Обработка DACL системой защиты при каждом использовании описателя процессом была бы неэффективной, поэтому SRM проверяет права доступа только при открытии описателя, а не при каждом его использовании. Так что, если процесс один раз успешно открыл описатель, система защиты не может аннулировать предоставленные при этом права доступа даже когда DACL объекта изменяется. Поскольку код режима ядра обращается к объектам по указателям, а не по описателям, при использовании объектов операционной системой права доступа не проверяются. Иначе говоря, исполнительная система Windows 2000 полностью доверяет себе в смысле защиты.

Тот факт, что владелец объекта всегда получает право на запись DACL при доступе к объекту, означает, что пользователям нельзя запретить доступ к принадлежащим им объектам. Если в силу каких-то причин DACL объекта пуст (доступ запрещен), владелец все равно может открыть объект с правом записи DACL и применить новый DACL, определяющий нужные права доступа.

Привилегия захвата во владение также представляет собой мощное средство в арсенале тех учетных записей, которым она назначена (например, учетной записи администратора). Обладателю этой привилегии доступен любой объект в системе. Допустим, какой-то объект принадлежит другому пользователю и *явно* запрещает администратору любые виды доступа к нему. Обладая привилегией захвата во владение, администратор может открыть объект с правом «запись владельца» (write-owner permission) и сменить владельца на Administrator. После этого он может закрыть объект и повторно открыть его с правом доступа к DACL для записи, а затем изменить DACL так, чтобы учетной записи администратора был предоставлен полный доступ к объекту.

5.2.7 Аудит безопасности

События аудита может генерировать диспетчер объектов в результате проверки прав доступа. Их могут генерировать и непосредственно Win32-функции, доступные пользовательским приложениям. Это же право, разумеется, есть и у кода режима ядра. С аудитом связаны две привилегии, SeSecurityPrivilege и SeAuPrivilege. Для управления файлом журнала безопасности, а также для просмотра и изменения SACL объектов процесс должен обладать привилегией SeSecurityPrivilege. Однако процесс, вызывающий системные сервисы аудита, должен обладать привилегией SeAuditPrivilege, чтобы успешно сгенерировать запись аудита в этом журнале.

Решения об аудите конкретного типа событий безопасности принимаются в соответствии с политикой аудита локальной системы. Политика аудита, также называемая локальной политикой безопасности (local security policy), является частью политики безопасности, поддерживаемой Lsass в локальной системе. При инициализации системы и изменении политики Lsass посылает SRM сообщения, информирующие его о текущей политике аудита. Lsass отвечает за прием записей аудита, генерируемых на основе событий аудита, обеспечивает их редактирование и передачу Event Logger (регистратору событий). Эти записи посылает именно Lsass (а не SRM), так как он добавляет в них сопутствующие подробности, например информацию, нужную для более полной идентификации процесса, по отношению к которому проводится аудит.

SRM посылает записи аудита Lsass через свое LPC-соединение. После этого Event Logger заносит записи в журнал безопасности. В дополнение к записям аудита, передаваемым SRM, Lsass и SAM тоже генерируют записи аудита, которые Lsass пересылает непосредственно Event Logger. Эта схема представлена на рисунке 5.4.

Записи аудита, подлежащие пересылке LSA, помещаются в очередь по мере получения — они не передаются пакетами. Пересылка этих записей осуществляется одним из двух способов. Если запись аудита невелика (меньше максимального размера LPC-сообщения), она посылается как LPC-сообщение. Записи аудита копируются из адресного пространства SRM в адресное пространство процесса Lsass. Если запись аудита велика, SRM делает ее доступной Lsass через разделяемую память и передает Lsass указатель на нее, используя для этого LPC-сообщение.

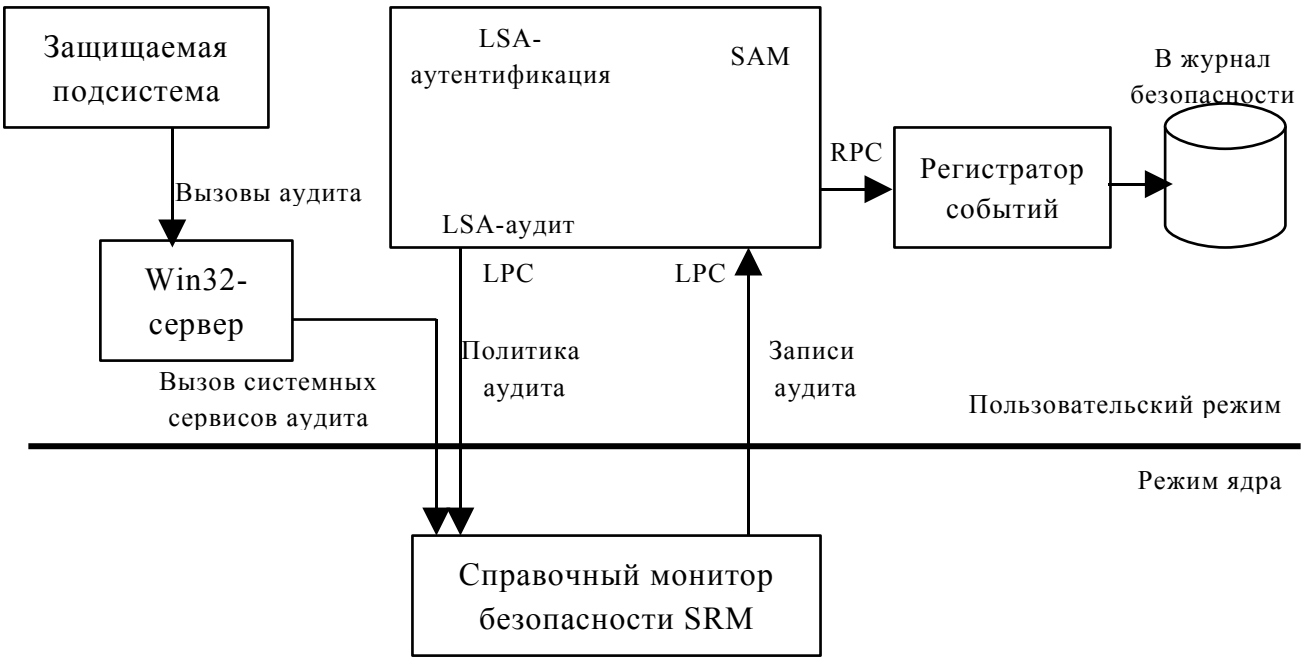

Рисунок 5.4

5.2.8 Работа с пользователями

При интерактивной регистрации в системе (в отличие от регистрации через сеть) происходит взаимодействие с процессами Winlogon, Lsass, одним или несколькими пакетами аутентификации, а также SAM или Active Directory. *Пакеты аутентификации* (authentication packages) — это DLL-модули, выполняющие проверки, связанные с аутентификацией. Пакетом аутентификации Windows 2000 для интерактивной регистрации в домене является Kerberos, аналогичным пакетом для интерактивной регистрации на локальных компьютерах, доменного входа в доверямыс домены под управлением версии Windows до Windows 2000, а также для регистрации в отсутствие контроллера домена -MSV1_0.

Winlogon является доверяемым процессом, отвечающим за управление взаимодействием с пользователем в связи с защитой. Он координирует регистрацию, запускает пользовательскую оболочку для входа в систему, обрабатывает выход из системы и управляет множеством других операций, имеющих отношение к защите, — вводом паролей

при регистрации, сменой паролей, блокированием и разблокированием рабочих станций и т. д. Процесс Winlogon должен обеспечить невидимость операций, связанных с защитой, другим активным процессам. Так, Winlogon гарантирует, что в ходе этих операций недоверяемый процесс не сможет перехватить управление рабочим столом и таким образом получить доступ к паролю.

Winlogon получает имя и пароль пользователя через Graphical Identification and Authentication (GINA) DLL Стандартная GINA—\\Winnt\System32\Msgina.dll. Msgina выводит диалоговое окно для входа в систему. Позволяя заменять Msgina другими GINA-библиотеками, Windows 2000 дает возможность менять механизмы идентификации пользователей, Например, сторонний разработчик может создать GINA для поддержки устройства распознавания отпечатков пальцев и для выборки паролей пользователей из зашифрованной базы данных.

Winlogon — единственный процесс, который перехватывает запросы на регистрацию с клавиатуры. Получив имя и пароль пользователя от GINA, Winlogon вызывает Lsass для аутентификации этого пользователя. Если аутентификация прошла успешно, процесс Winlogon активизирует оболочку. Схема взаимодействия между компонентами, участвующими в процессе регистрации, показана на рисунке 5.5.

Winlogon не только поддерживает альтернативные GINA, но и может загружать дополнительные DLL компонентов доступа к сетям, необходимые для вторичной аутентификации. Это позволяет сразу нескольким компонентам сетевого доступа получать идентификационные и регистрационные данные в процессе обычного входа пользователя в систему. Например, регистрируясь в системе под управлением Windows 2000, пользователь может одновременно зарегистрироваться и на UNIX-сервере. После этого он получит доступ к ресурсам UNIX- сервера с компьютера под управлением Windows 2000 без дополнительной аутентификации.

При инициализации системы Winlogon выполняет ряд операций, обеспечивающих ему контроль над рабочей станцией с момента готовности системы к взаимодействию с пользователем:

1) создает и открывает интерактивный объект WindowStation, представляющий клавиатуру, мышь и монитор. Далее создает дескриптор защиты станции с одним ACE, содержащим только SID-идентификатер Winlogon. Этот

уникальный дескриптор безопасности гарантирует, что другой процесс получит доступ к рабочей станции, только если Winlogon явно разрешит это;

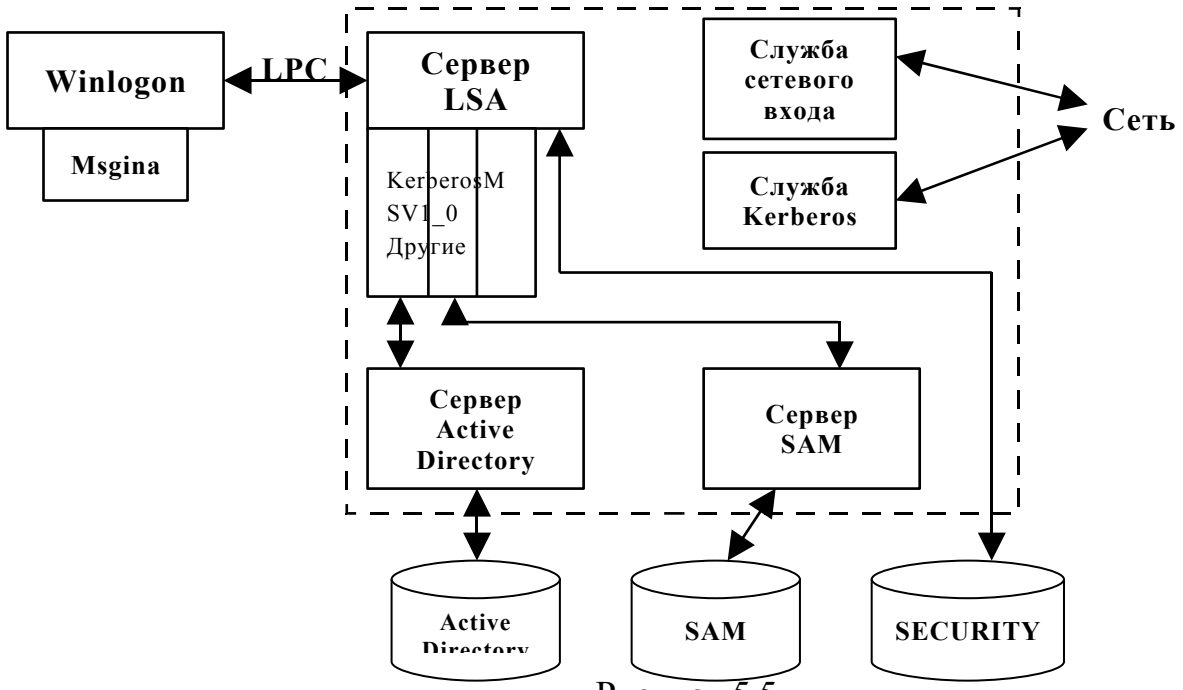

Рисунок 5.5

2) создает и открывает три объекта «рабочий стол»: для приложений, Winlogon и экранной заставки. Защита объекта «рабочий стол» Winlogon организуется так, чтобы к нему мог обращаться только Winlogon. Другие объекты «рабочий стол» доступны как Winlogon, так и пользователям. Следовательно, пока активен объект «рабочий стол» Winlogon, никакой другой процесс не получает доступа к коду и данным, сопоставленным с этим рабочим столом. Эта функциональность используется Windows 2000 для защиты операций, требующих передачи паролей, а также для блокировки и разблокировки рабочего стола.

До регистрации какого-либо пользователя на компьютере видимым рабочим столом является объект «рабочий стол» Winlogon. После регистрации нажатие клавиш Ctrl+Alt+Del вызывает переключение объектов «рабочий стол» – с Default на Winlogon. Таким образом, SAS всегда активизирует защищенный объект «рабочий стол», контролируемый Winlogon;

3) устанавливает LPC-соединение с Lsass через порт LsaAuthenticationPort (вызовом *LsaRegisterLogonProcess).* Это соединение понадобится для обмена информацией при входе и выходе пользователя из системы при операциях с паролем. Далее \Winlogon настраивает оконную среду;

4) инициализирует и регистрирует структуру данных оконного класса, которая сопоставляет процедуру Winlogon с создаваемым ею окном.

5) регистрирует SAS, сопоставляя ее с только что созданным окном. Это гарантирует, что ввод пользователем SAS будет вызывать именно оконную процедуру Winlogon и что программы типа троянских коней не смогут перехватывать управление при вводе SAS;

6) регистрирует окно, чтобы при выходе пользователя вызывалась процедура, сопоставленная с этим окном. Подсистема Win32 проверяет, что запросивший уведомление процесс является именно Winlogon.

Как только при инициализации системы создается рабочий стол Winlogon, он становится активным рабочим столом. Причем активный рабочий стол Winlogon всегда заблокирован. Winlogon разблокирует свой рабочий стол лишь для переключения на рабочий стол приложений или экранной заставки (блокировать или разблокировать рабочий стол может только процесс Winlogon.)

Регистрация пользователя начинается после нажатия комбинации клавиш SAS (по умолчанию — Ctrl+Alt+Del). После этого Winlogon вызывает GINA, чтобы получить имя и пароль пользователя. Winlogon также создает уникальную локальную группу для этого пользователя и назначает ее данному экземпляру объекта «рабочий стол» (который представляет клавиатуру, экран и мышь), Winlogon передает имя этой группы и Lsass при вызове *LsaLogonUser***.** Если регистрация пользователя прошла успешно, эта группа будет включена в маркер процесса регистрации (logon process marker) — такой шаг предпринимается для защиты доступа к объекту «рабочий стол». Например, второй пользователь, зарегистрировавшийся по той же учетной записи, но в другой системе, не получит доступа для записи к объекту «рабочий стол» первого пользователя, так как не входит в его группу.

После ввода имени и пароля пользователя Winlogon поочередно вызывает все пакеты аутентификации, зарегистрированные в системе. Winlogon получает описатель пакета вызовом Lsass-функции *LsaLookupAutbenticationPackage* и передает пакету регистрационные данные через *LsaLogon User.* После того как пакет аутентифицирует пользователя, Winlogon продолжает процесс регистрации этого пользователя. Если ни один из пакетов не сообщает об успешной аутентификации, регистрация прекращается.

Windows 2000 использует два стандартных пакета аутентификации: Kerberos и MSV1 _0. Пакет аутентификации по умолчанию в автономной

системе Windows 2000 — MSV1_0 (\Winnt\System32\Msvl_0.dll. Lsass также использует MSV1_0 на компьютерах, входящих в домен, чтобы аутентифицировать домены и компьютеры под управлением версий Windows до Windows 2000, не способные найти контроллер домена для аутентификации. Пакет аутентификации Kerberos (\Winnt \System32\Kerberos.dll) используется на компьютерах, входящих в домены Windows 2000. Этот пакет во взаимодействии со службами Kerberos, выполняемыми на контроллере домена, поддерживает протокол Kerberos версии 5 (ревизии 6).

Пакет аутентификации MSV1_0 принимает имя пользователя и хэшированную версию пароля и посылает локальному SAM запрос на получение информации из учетной записи, включая пароль, группы, в которые входит пользователь, и список ограничений по данной учетной записи. Сначала MSV1_0 проверяет ограничения, например, разрешенное время или типы доступа. Если ограничения из базы данных SAM запрещают регистрацию пользователя в это время суток, MSV1_0 возвращает LSA статус отказа.

Далее MSVl_0 сравнивает хэшированный пароль и имя пользователя с теми, которые хранятся в SAM. В случае хэшированной доменной регистрации MSV1_0 обращается к квитированной информации через функции Lsass, отвечающие за сохранение и получение «секретов» из базы данных LSA (куст реестра SECURITY). Если эти данные совпадают, MSV1_0 генерирует LUID сеанса регистрации и создает собственно сеанс регистрации вызовом Lsass. При этом MSV1_0 сопоставляет данный уникальный идентификатор с сеансом и передает данные, необходимые для того, чтобы в конечном счете создать маркер доступа для пользователя. Если MSV1_0 требуется аутентифицировать пользователя с удаленной системы, например, при его регистрации в доверяемом домене под управлением версий Windows до Windows 2000, то MSV1_0 взаимодействует с экземпляром Netlogon в удаленной системе через службу NetLogon (сетевого входа в систему). Netlogon в удаленной системе взаимодействует с пакетом аутентификации MSVl_0 этой системы, передавая результаты аутентификации системе, в которой проходит регистрация,

Базовая последовательность действий при аутентификации Kerberos в основном та же, что и в случае MSV1_0. Однако в большинстве случаев доменная регистрация проходит на рабочих станциях или серверах,

входящих в домен (а не на контроллере домена), поэтому пакет в процессе аутентификации должен взаимодействовать с ними через сеть. Взаимодействие этого пакета со службой Kerberos на контроллере домена осуществляется через TCP/IP-порт Kerberos

Служба Kerberos (\Winnt\System32 \Kdcsvcdll), реализующая протокол аутентификации Kerberos, выполняется в процессе Lsass на контроллерах домена.

После проверки хэшированной информации об имени и пароле пользователя с помощью объектов учетных записей пользователей возвращает удостоверения домена Lsass, который при успешной регистрации передает через сеть результат аутентификации и удостоверения доменной регистрации пользователя той системе, где проходит регистрация.

6 Защита информации в вычислительных сетях

6.1 Защита данных с использованием сервисов IPSec

Одним из средств защиты, применяемым на транспортном уровне OSI, является использование криптографических средств защиты информации. Рассмотрим наиболее распространенный на сегодняшний день протокол защиты IP – IPSec. [20, 39]

Защита на уровне IP охватывает три сферы безопасности: аутентификацию, конфиденциальность и управление ключами. Механизм аутентификации гарантирует, что полученный пакет был на самом деле отправлен стороной, указанной как источник в заголовке пакета. Кроме того, этот механизм гарантирует то, что пакет не был изменен в процессе его доставки. Средства конфиденциальности обеспечивают возможность взаимодействующим сторонам шифровать сообщения, что позволяет исключить ознакомление третьих лиц с передаваемой информацией. Средства управления ключами гарантируют защищенный обмен ключами.

Отметим, что требования средств аутентификации и шифрования как необходимых средств защиты включены в требования стандарта протокола IPv6, однако эти средства допускают применение и в действующем протоколе IPv4

Протокол IPSec обеспечивает защищенный обмен данными в локальных сетях, корпоративных и открытых глобальных сетях и Internet. Главным свойством IPSec является возможность шифрования и (или) аутентификации всего потока данных на уровне IP. Это позволяет обеспечить защиту практически любых приложений, в том числе и web-сервисов.

Преимуществами IPSec являются:

–обеспечение надежной защиты, которая может быть применена ко всему потоку данных, адресованных за пределы локальной сети, в случае встраивания IPSec в брандмауэр или маршрутизатор;

–IPSec в брандмауэре трудно обойти, если он является единственной точкой связи локальной сети и Internet, и весь поток входящих данных должен использовать IP;

–прозрачность для приложений, так как расположение IPSec на транспортном уровне позволяет не модифицировать программное обеспечение верхнего уровня;

–возможность скрытия IPSec от конечного пользователя.

Кроме того, IPSec обеспечивает защиту маршрутизации, т.е. гарантирует, что:

–извещение о появлении нового маршрутизатора приходит от уполномоченного маршрутизатора;

–извещение соседнего маршрутизатора при взаимодействии с маршрутизатором в другом домене маршрутизации приходит от уполномоченного маршрутизатора;

–сообщение переадресации приходит именно от того маршрутизатора, которому изначально посылался пакет;

–обновление маршрута не является фальсификацией.

IPSec обеспечивает сервис защиты на уровне IP, позволяя системе выбрать необходимые протоколы защиты, определить алгоритмы для соответствующих сервисов и задать значения криптографических ключей, необходимых для запрошенных сервисов. Для защиты используется два протокола: протокол аутентификации и протокол шифрования-аутентификации. Средства защиты реализуются в виде заголовков расширений, которые следуют за основным заголовком IP. Заголовок расширения аутентификации называется заголовком AH (Authentication Header, заголовок аутентификации), заголовок расширения шифрования – заголовком ESP (Encapsulating Security Payload Header, заголовок включающего защиту полезного груза). Как AH, так и ESP имеют возможности контроля доступа, основанного на распределении криптографических ключей и управлении транспортными потоками, относящимися к этим протоколам защиты. Сервисы, обеспечиваемые IPSec, показаны в таблице 6.1.

Таблица 6.1 – Сервисы IPSec

Сервис	AH	ESP (только шифрование)	ESP (шифрование и аутентификация)
Контроль доступа	X	X	X
Целостность без установления соединений	X		X
Аутентификация источника данных	X		X
Отторжение воспроизведенных пакетов	X	X	X
Конфиденциальность		X	X
Ограниченная конфиденциальность транспортного потока		X	X

Базовым объектом в механизмах аутентификации и конфиденциальности для IPSec является защищенная связь (Security Association). Связь представляет собой одностороннее отношение между отправителем и получателем, применяющим сервис защиты к транспортному потоку. Если требуется равноправное отношение для двустороннего защищенного обмена, необходимы две защищенные связи. Сервис защиты предоставляет возможность для защищенной связи использовать либо AH, либо ESP.

Защищенная связь однозначно определяется следующими тремя параметрами:

–индекс параметров защиты – битовая строка, присваиваемая данной защищенной связи и имеющая только локальное значение. Индекс параметров защиты используется, чтобы дать принимающей стороне возможность выбрать защищенную связь, по которой должен обрабатываться полученный пакет;

–адрес IP-пакета. В настоящее время допускаются только однонаправленные адреса – адрес пункта назначения в защищенной связи (конечная система либо сетевой объект);

–идентификатор протокола защиты – указывает, является связь защищенной связью AH или ESP.

Таким образом, в любом IP-пакете защищенная связь однозначно идентифицируется адресом пункта назначения в

заголовке IP и индексом параметров защиты во вложенном заголовке расширения AH или ESP.

В каждой реализации IPSec имеется таблица защищенных связей (Security Association Database), которая определяет параметры, связанные с каждой защищенной связью. Защищенная связь обычно имеет следующие параметры:

–счетчик порядкового номера – 32-битовое значение, необходимое для поля порядкового номера в заголовках AH или ESP;

–флаг переполнения счетчика порядкового номера – указывает на переполнение счетчика порядкового номера. При переполнении счетчика прекращается дальнейшая передача пакетов с применением этой защищенной связи;

–окно защиты от воспроизведения – служит для определения, является ли прибывший AH или ESP пакет воспроизведением;

–информация AH – алгоритм аутентификации, ключи, продолжительности жизни ключей и другие параметры, используемые AH;

–информация ESP – алгоритм шифрования и аутентификации, ключи, значения инициализации, продолжительности жизни ключей и другие параметры, используемые ESP;

–продолжительность жизни защищенной связи – интервал времени или значение счетчика байтов, после которого защищенная связь должна быть заменена новой защищенной связью с новым индексом параметра защиты или завершена с указанием того, какое действие должно при этом произойти;

–режим протокола – указывает на использование туннельного, транспортного или задаваемого групповым символом режима;

–максимальная единица передачи (MTU, Maximum Transmission Unit) маршрута – максимальная единица передачи для любого допустимого маршрута и переменные времени существования.

Механизм управления ключами связывается с механизмами аутентификации и конфиденциальности только через индекс параметров защиты. Таким образом, аутентификация и

конфиденциальность оказываются определенными независимо от конкретного механизма управления ключами.

IPSec обеспечивает значительную гибкость в выборе способа применения сервиса IPSec к протоколу IP. Кроме того, IPSec различает трафик, подлежащий защите IPSec, и трафик, которому позволяется обойти IPSec.

AH и ESP поддерживают два режима использования: транспортный и туннельный. Схема работы протокола IPSec показана на рисунке 6.1.

Транспортный режим обеспечивает защиту для протоколов высшего уровня (например, TCP, UDP, ICMP), т.е. в этом режиме защита распространяется на полезный груз пакета IP. В большинстве случаев транспортный режим обеспечивает сквозную связь двух главных узлов (например, клиента и сервера). Для протокола IPv4 полезным грузом являются данные, находящиеся после заголовка IP. Для IPv6 полезным грузом являются данные, следующие за заголовком IP и всех имеющихся расширений IPv6 (возможна также защита заголовка параметров адресата). ESP в транспортном режиме шифрует и идентифицирует (в случае необходимости) только полезный груз IP, исключая заголовок. AH в транспортном режиме идентифицирует полезный груз IP и некоторые части заголовка IP.

Туннельный режим обеспечивает защиту всего пакета IP. При этом после добавления к IP-пакету полей AH или ESP весь пакет вместе с полями защиты рассматривается как полезный груз некоторого нового IP-пакета с новым заголовком IP. Весь оригинальный пакет при этом пересылается от одной точки сети к другой, и ни один из маршрутизаторов не может проверить внутренний заголовок IP. Обычно туннельный режим используется для связи между узлами, являющимися шлюзами защиты (например, маршрутизаторами с IPSec). В этом случае системы, расположенные в сетях за брандмауэрами, могут обмениваться между собой незащищенными IP-пакетами и с некоторой другой подсетью через защищенный туннель в глобальных сетях.

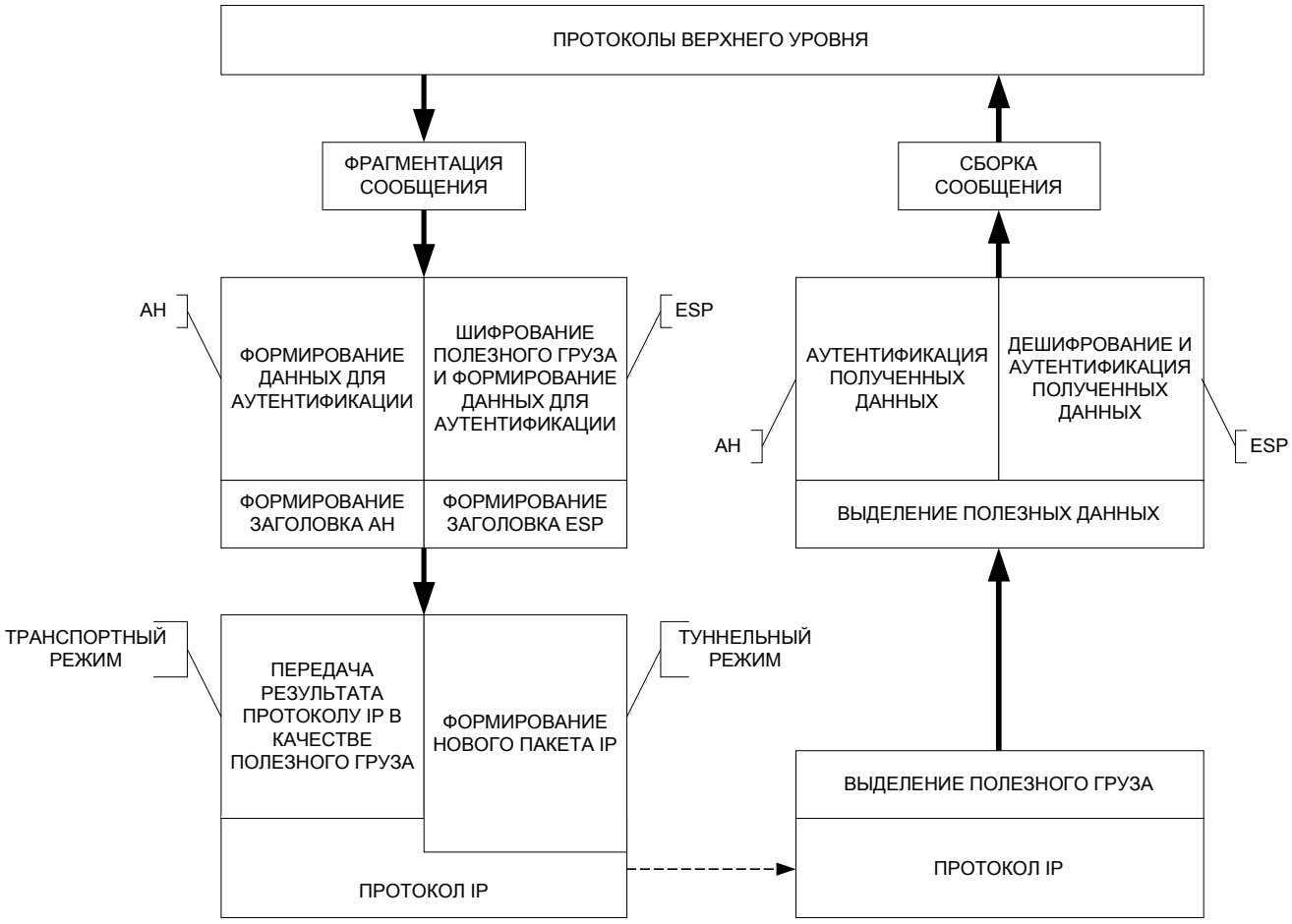

Рисунок 6.1 – Работа протокола IPSec

Заголовок аутентификации AH обеспечивает поддержку целостности данных и аутентификации пакета IP. Свойство целостности данных гарантирует невозможность несанкционированной модификации содержимого пакета на пути следования. Функция аутентификации дает возможность конечной системе или сетевому устройству идентифицировать пользователя или приложение, обеспечивая таким образом защиту от атак через Internetc подменой сетевых адресов.

Сервис AH использует код аутентичности сообщения (MAC, Message Authentication Code), поэтому обе стороны должны использовать общий секретный ключ. Все существующие реализации поддерживают две схемы получения кода контроля целостности: HMAC-MD5-96 и HMAC-SHA-1-96 (с использованием алгоритмов MD5 и SHA-1 соответственно). В обеих ситуациях вычисляется полное значение HMAC, однако в дальнейшем используются только первые 96 битов.

Для вычисления значения MAC используются неизменяемые поля IP (например, поле адреса источника или длины заголовка для IPv4), заголовок AH, отличный от поля данных аутентификации, и все данные протокола верхнего уровня, неизменные в пути следования (например, сегмент TCP).

Для защиты от воспроизведения пакетов используется поле порядкового номера. При установлении новой защищенной связи отправитель инициализирует счетчик порядкового номера, устанавливая его значение равным нулю. При передаче по этой защищенной связи пакета отправитель увеличивает значение счетчика на единицу и помещает это значение в поле порядкового номера. Отправитель не должен допускать порядковые номера, превышающие $2^{32} – 1$. При достижении этого предела отправитель завершает данную защищенную связь и начинает переговоры о создании новой защищенной связи с новым ключом.

Протокол IP не гарантирует, что пакеты будут доставлены в требуемом порядке, а также не гарантирует доставку всех пакетов. Для обеспечения этого IPSec использует окно, правая граница которого представляет наивысший порядковый номер до сих пор полученных действительных пакетов. Обработка принятого пакета выполняется следующим образом:

–если полученный пакет попадает в рамки окна и является новым, то проверяется значение MAC. В случае успешной аутентификации помечается сегмент в окне;

–если полученный пакет попадает в область правее окна и является новым, то проверяется значение MAC. Если аутентификация успешна, то окно раздвигается так, чтобы новый порядковый номер оказался правой границей окна, а в окне помечается соответствующий фрагмент;

–если полученный пакет оказывается слева от окна или аутентификация завершается неудачей, пакет отвергается.

Протокол ESP обеспечивает сервис конфиденциальности, в том числе защиту содержимого сообщения и ограниченную конфиденциальность для транспортного потока. Кроме того, ESP может обеспечить тот же сервис аутентификации, что и AH.

Сервис ESP предполагает шифрование полезного полей груза, заполнителя, длины заполнителя и следующего заголовка. Если для используемого при шифровании полезного груза алгоритма требуется криптографическая синхронизация, например, вектор инициализации (IV), то необходимые данные могут быть перенесены в начало поля полезного груза. При наличии значения IV оно обычно не шифруется, хотя считается частью шифрованного текста.

Спецификации IPSec требуют наличия поддержки алгоритма DES в режиме сцепления шифрованных блоков (CBC) во всех реализациях. Кроме того, возможно использование других алгоритмов:

–тройной DES;

–RC5;

–IDEA;

–тройной IDEA;

–Blowfish.

Как и AH, протокол ESP поддерживает использование MAC длиной 96 битов, поддерживает схемы HMAC-MD5-96 и HMAC-SHA-1-96.

Отдельная защищенная связь может использовать протокол AH или ESP, однако иногда конкретный поток обмена данными может требовать как сервиса AH, так и сервиса ESP. Кроме того, потоку обмена данными может потребоваться сервис IPSec для связи между главными узлами и другой сервис для связи между шлюзами защиты. В этих случаях используются т.н. пучки защищенных связей (Security Association Bundle) – набор защищенных связей, посредством которых потоку предоставляется необходимое множество услуг IPSec. При этом защищенные связи в пучке могут завершаться в различных конечных точках.

Существуют два способа объединения защищенных связей в пучки:

–транспортная смежность – применение более одного протокола защиты к одному пакету IPSec без туннелирования. Данный подход эффективен только для одного уровня вложенности;

–повторное туннелирование – применение нескольких уровней протоколов защиты с помощью туннелирования IP. Этот подход допускает множество уровней вложения, поскольку туннели могут начинаться и завершаться в разных использующих IPSec узлах сети вдоль маршрута передачи данных.

Шифрование и аутентификация могут комбинироваться для того, чтобы обеспечить передаваемому IP-пакету как конфиденциальность, так и аутентификацию. Протокол IPSec позволяет реализовать несколько подходов к такому комбинированию.

В случае использования ESP с опцией аутентификации пользователь сначала применяет ESP к требующим защиты данным, а затем добавляет поле аутентификации. При этом можно выделить две возможности:

–транспортный режим ESP, при котором аутентификация и шифрование применяются к полезному грузу IP, однако не защищается заголовок IP;

–туннельный режим ESP, при котором аутентификация применяется ко всему пакету IP, доставляемому по адресу IP внешнего получателя (например, брандмауэра), и аутентификация выполняется в системе адресата. Таким образом, защищается весь внутренний пакет IP.

В обоих случаях аутентификации подлежит шифрованный текст.

Другим способом применения аутентификации после шифрования является использование пучка из двух транспортных защищенных связей, где внутренняя связь является ESP-связью, а внешняя – AH-связью. В этом случае ESP используется без аутентификации, и шифрование применяется к полезному грузу IP. Получаемый в результате пакет состоит из IP-заголовка, за которым следуют данные ESP, затем применяется AH в транспортном режиме. Таким образом, аутентификация будет охватывать данные ESP и оригинальный IP-заголовок за исключением изменяемых полей.

В некоторых случаях может быть предпочтительным использование функций аутентификации до шифрования. Например, это позволяет защитить данные аутентификации от

изменения, так как они будут защищены шифрованием. Кроме того, может быть желательным сохранение информации аутентификации вместе с сообщением для дальнейшего использования. Вариантом применения аутентификации до шифрования является использование пучка защищенных связей, состоящего из внутренней транспортной AH-связи и внешней туннельной ESP-связи. В этом случае аутентификация охватывает полезный груз IP вместе с заголовком (за исключением изменяемых полей). Полученный таким образом пакет обрабатывается в туннельном режиме ESP, в результате чего весь внутренний пакет является шифрованным и имеющим новый внешний IP-заголовок.

Функции управления ключами IPSec включают создание и распределение секретных ключей. Для защищенной связи между двумя приложениями обычно требуются четыре ключа: по одной паре ключей для передачи и приема данных как для AH, так и для ESP. Стандарт протокола предусматривает два типа управления ключами: ручное и автоматизированное. В первом случае системный администратор вручную конфигурирует каждую систему с указанием требуемых ключей и необходимых ключей других сообщающихся систем. В случае автоматизированного управления система обеспечивает возможность создания ключей для защищенных связей по запросу и, таким образом, упрощает проблему использования ключей в большой распределенной системе с изменяющейся конфигурацией.

По умолчанию в IPSec применяется протокол автоматизированного управления ключами ISAKMP/Oakley, состоящий из двух компонентов: протокола определения ключей и протокола управления ключами.

Протокол определения ключей Oakley представляет собой протокол обмена ключами, основанный на алгоритме Диффи-Хеллмана, но обеспечивающий дополнительную защиту. Протокол Oakley не требует использования каких-либо определенных форматов.

Протокол управления ключами ISAKMP обеспечивает основу схемы управления ключами в Internet и поддержку некоторого протокола наряду с соответствующими форматами для

согласования атрибутов защиты. ISAKMP предлагает множество типов сообщений, которые могут использоваться с рядом алгоритмов обмена ключами. Oakley является конкретным алгоритмом обмена ключами, предназначенным для использования с исходной версией ISAKMP.

Алгоритм Oakley поддерживает использование различных групп обмена ключами Диффи-Хеллмана. В каждой группе определяются два глобальных параметра и природа алгоритма. На сегодняшний день возможно использование следующих групп:

–возведение в степень в арифметике классов вычетов с модулем длиной 768 битов;

–возведение в степень в арифметике классов вычетов с модулем длиной 1024 бита;

–возведение в степень в арифметике классов вычетов с 1024-битовым модулем и произвольно определяемыми параметрами;

–группа эллиптической кривой над конечным полем из 2^{155} элементов;

–группа эллиптической кривой над конечным полем из 2^{185} элементов.

С алгоритмом Oakley могут применяться различные методы аутентификации: цифровые подписи, шифрование с открытым ключом и шифрование с симметричным ключом. В первом случае аутентификация обмена данными осуществляется с помощью подписи взаимно доступного хеш-кода: каждая сторона шифрует хеш-код своим личным ключом. Во втором случае аутентификация обмена данными осуществляется с помощью шифрования параметров, например таких, как идентификаторы пользователей, и использованием личного ключа отправителя. В случае шифрования с симметричным ключом для аутентификации обмена данными может использоваться шифрование параметров обмена по симметричной схеме с помощью некоторого ключа, получаемого с применением какого-либо дополнительного механизма.

Протокол ISAKMP определяет процедуры и форматы пакета, предназначенные для использования в процессе переговоров о создании, изменении или удалении защищенных связей. Как часть процесса создания защищенной связи ISAKMP определяет полезный груз сообщений обмена ключами и аутентификации

данных. Форматы полезного груза обеспечивают основу, не зависящую от конкретного протокола обмена ключами, алгоритма шифрования или механизма аутентификации.

Сообщение ISAKMP состоит из заголовка ISAKMP и следующих за ним полезных грузов. Сообщение передается с помощью какого-либо транспортного протокола. Действующие спецификации требуют, чтобы любая реализация поддерживала использование протокола UDP в качестве транспортного.

В таблице 6.2 приведены типы полезного груза, допустимые для ISAKMP, и указаны поля или параметры, составляющие полезный груз каждого из этих типов.

Таблица 6.2 – Типы полезного груза ISAKMP

Тип	Параметры	Описание
1	2	3
Защищенная связь (SA)	Область интерпретации, ситуация	Используется для согласования атрибутов защиты и указания области интерпретации и ситуации, при которых выполняется такое согласование
Предложение (P)	Номер предложения, идентификатор протокола, длина индекса параметров защиты, число трансформаций, индекс параметров защиты	Используется в ходе согласования параметров создаваемой защищенной связи, указывает применяемый протокол и число трансформаций
Трансформация	Номер трансформации,	Применяется в ходе согласования
(T)	идентификатор трансформации, атрибуты защищенной связи	параметров защищенной связи, указывает преобразование и соответствующие атрибуты защищенной связи

Продолжение таблицы 6.2

1	2	3
Обмен ключами (KE)	Данные обмена ключами	Поддерживает ряд методов обмена ключами

Идентификация (ID)	Тип идентификации, данные идентификации	Предназначен для обмена информацией идентификации
Сертификат (CERT)	Кодировка сертификата, данные сертификата	Служит для пересылки сертификатов и другой связанной с сертификатами информации
Запрос сертификата (CR)	Число типов сертификатов, типы сертификатов, число центров сертификации, центры сертификации	Используется для запросов сертификатов, указывает типы запрашиваемых сертификатов и приемлемые центры сертификации
Хеширование (HASH)	Данные хеширования	Содержит данные, генерируемые функцией хеширования
Подпись (SIG)	Данные подписи	Содержит данные, генерируемые функцией цифровой подписи
Оказия (NONCE)	Данные оказии	Содержит локально порожденное случайное число, используемое для защиты от воспроизведения
Уведомление (N)	Область идентификации, идентификатор протокола, длина индекса параметров защиты, тип уведомления сообщения, индекс параметров защиты, данные уведомления	Используется для передачи данных уведомления, например, признака возникновения ошибки

Продолжение таблицы 6.2

1	2	3
Удалить (D)	Область идентификации, идентификатор протокола, длина индекса параметров	Указывает защищенную связь, которая более не является действительной

	защиты, число индексов параметров защиты, индекс параметров защиты (один или более)	

Полезный груз типа SA служит для того, чтобы начать процесс создания защищенной связи. В этом случае параметр области интерпретации идентифицирует область интерпретации (DOI), с использованием которой выполняется согласование. Параметр ситуации определяет политику защиты для такого процесса согласования, устанавливая уровни защиты, требуемые для шифрования и конфиденциальности.

Полезный груз типа P включает информацию, используемую в ходе согласования атрибутов создаваемой защищенной связи, а также указывает протокол для этой защищенной связи (ESP или AH), в отношении которого ведутся переговоры о соответствующих сервисах и механизмах. Полезный груз этого типа также включает индекс параметров защиты объекта-отправителя и число трансформаций. Каждая трансформация содержится в полезном грузе типа T. Применение нескольких полезных грузов типа трансформации позволяет инициатору предложить на выбор несколько возможностей, из которых отвечающая сторона должна выбрать одну или отвергнуть предложение.

Полезный груз типа T определяет защитную трансформацию, которая должна использоваться для защиты коммуникационного канала в соответствии с указанным протоколом. Номер трансформации позволяет идентифицировать данный полезный груз с тем, чтобы отвечающая сторона могла сослаться на этот номер для указания своего решения использовать данную трансформацию. Поля идентификатора трансформации и атрибутов идентифицируют саму трансформацию (например, 3DES для ESP или HMAC-SHA-1-96 для AH) и связанные с ней параметры (например, длину хеш-кода).

Полезный груз типа KE может использовать ряд методов обмена ключами, включая Oakley, обмен по Диффи-Хеллману и

обмен ключами RSA, предусмотренный в PGP. Ключевое поле данных обмена включает данные, требуемые, чтобы создать ключ сеанса, и зависит от используемого алгоритма обмена.

Полезный груз типа ID предназначен для идентификации сообщающихся сторон и может служить для проверки аутентичности информации. Обычно поле данных идентификации содержит адрес IPv4 или IPv6.

Полезный груз типа CERT содержит сертификат открытого ключа. Поле кодировки сертификата указывает тип сертификата или связанную с сертификатом информацию, которая может включать:

–сертификат X.509 в формате KCS #7;

–сертификат PGP;

–подписанный ключ DNS;

–сертификат X.509 – подпись;

–сертификат X.509 – обмен ключами;

–билеты (tickets) Kerberos;

–список отозванных сертификатов (CRL);

–список отозванных полномочий (ARL);

–сертификат SPKI.

В любой точке обмена ISAKMP отправитель может разместить полезный груз типа CR, чтобы запросить сертификат второй из участвующих в обмене сторон. Этот полезный груз может содержать список нескольких типов сертификатов и нескольких центров сертификации, которые рассматриваются как приемлемые.

Полезный груз типа HASH содержит данные, генерируемые функцией хеширования для некоторой части сообщения и (или) состояния ISAKMP. С помощью этого полезного груза можно проверить целостность данных в сообщении или аутентифицировать объекты, ведущие переговоры.

Полезный груз типа SIG содержит данные, генерируемые функцией цифровой подписи для некоторой части сообщения и (или) состояния ISAKMP. Этот полезный груз служит для проверки целостности данных в сообщении и может использоваться для сервиса, гарантирующего невозможность отказа от сообщений.

Полезный груз типа NONCE включает случайные данные, обеспечивающие гарантию выполнения процесса обмена в реальном времени и защиту его от атак воспроизведения сообщений.

Полезный груз типа N содержит информацию об ошибке или состоянии, связываемом с данной защищенной связью или данным процессом согласования параметров защищенной связи. Помимо сообщений об ошибках ISAKMP и сообщения состояния Connected для ISAKMP для IPSec определяются следующие дополнительные сообщения состояния:

Responder-Lifetime – продолжительность жизни защищенной связи, выбранная отвечающей стороной;

Replay-Status – служит для подтверждения положительного решения отвечающей стороны по вопросу о применении средств обнаружения атак воспроизведения;

Initial-Contact – информирует другую сторону о том, что данная защищенная связь является первой защищенной связью, создаваемой с удаленной системой. Получатель такого уведомления может удалить все существующие защищенные связи с системой-отправителем, предполагая, что система отправителя была перезагружена и больше не имеет доступа к старым защищенным связям.

Полезный груз типа D указывает на одну или несколько защищенных связей, которые предполагаются недействительными ввиду того, что отправитель удалил их из своей базы данных.

Протокол ISAKMP обеспечивает основу для обмена сообщениями. Спецификации указывают пять типов обмена, которые должны поддерживаться по умолчанию.

Базовый обмен позволяет произвести обмен ключами и данными аутентификации одновременно. Это сводит к минимуму число сообщений обмена за счет отказа от обеспечения защиты идентификации сторон. Два первых сообщения обеспечивают маркеры и создают защищенную связь с согласованными протоколом и трансформациями. Обе стороны используют механизм защиты от атак воспроизведения. В следующих двух сообщениях происходит обмен относящимися к ключам данными и идентификаторами пользователей, а полезный груз AUTH

используется для проверки ключей и участвующих в обмене сторон.

Обмен с защитой идентификации сторон является расширением базового обмена с целью защиты информации об участвующих в обмене сторонах. Два первых сообщения создают защищенную связь, два следующих осуществляют обмен ключами. Как только становится доступным сеансовый ключ, стороны обмениваются шифрованными сообщениями, содержащими информацию аутентификации, например, цифровые подписи и в некоторых случаях сертификаты открытых ключей.

Обмен только данными аутентификации используется для осуществления взаимной аутентификации без обмена ключами. Два первых сообщения создают защищенную связь. Кроме того, отвечающая сторона использует второе сообщение для того, чтобы передать свой идентификатор, и прибегает к аутентификации, чтобы защитить это сообщение. Инициатор посылает третье сообщение, чтобы передать свой удостоверенный идентификатор.

Энергичный обмен сводит к минимуму число сообщений обмена за счет отказа от обеспечения защиты идентификации сторон. В первом сообщении инициатор предлагает создать защищенную связь с набором возможных протоколов и трансформаций. Кроме того, инициатор начинает обмен ключами и высылает свой идентификатор. Во втором сообщении отвечающая сторона указывает на принятие защищенной связи с уже конкретными протоколом и трансформацией, завершает обмен ключами и удостоверяет передаваемую информацию. В третьем сообщении инициатор передает результат аутентификации предыдущей информации, шифруя его с использованием общего сеансового ключа.

Информационный обмен предназначен для односторонней пересылки информации, касающейся параметров управления защищенной связью (например, уведомление об ошибке).

В настоящее время протокол IPSec поддерживается практически всеми операционными системами. Также имеются реализации IPSec, встроенные в различные сетевые устройства, например, маршрутизаторы или брандмауэры. В качестве примера можно привести маршрутизаторы D-Link DI804-HV и DI808-HV,

LYNKSYS RV082. В операционной системы Windows установка поддержки протокола IPSec сводится к установке и настройке сетевого компонента для протокола IPSec. В операционной системе Linux для поддержки протокола IPSec используется реализации IPSec, созданные в рамках проектов FreeSWAN (в настоящий момент проект приостановлен) или KAME.

Отметим, что защита уровня IP с помощью протокола IPSec применяется в основном для организации виртуальных частных сетей и объединения нескольких сегментов корпоративной сети через незащищенную среду передачи (например, через Internet).

6.2 Защита данных с использованием протоколов SSL/TLS

Для защиты данных в сети Internet на прикадном уровне модели OSI рекомендуется использование протоколов SSL и TLS (Secure Socket Layer и Transport Layer Security) [38, 44 – 48].

Протокол SSL был предложен компанией Netscape как проект стандарта защиты данных в Internet. В настоящее время используется протокол SSLv3, а также созданный рабочий группой TLS вариант стандарта TLSv1. Протокол TLS реализует все возможности SSLv3, за исключением поддержки алгоритмов обмена ключами Fortezza и алгоритмов симметричной схемы шифрования Fortezza. Кроме того, имеется отличие в наборе полей записи, по которым вычисляется значение MAC.

Протокол SSL предназначен для обеспечения надежной защиты сквозной передачи данных с использованием протокола TCP. Протокол SSL имеет двухуровневую структуру, показанную на рисунке 6.2.

Рисунок 6.2 – Структура протокола SSL

Протокол записи SSL (SSL Record Protocol) обеспечивает базовый набор средств защиты, применяемых протоколами более высоких уровней. Эти средства, в частности, может использовать протокол передачи гипертекста HTTP, обеспечивающий обмен данными при взаимодействии клиентов и серверов web. Частью SSL также считаются следующие протоколы более высокого уровня: протокол квитирования установления связи (SSL Handshake Protocol), протокол изменения параметров шифрования (SSL Change Cipher Spec Protocol) и протокол извещения (SSL Alert Protocol). Эти протоколы служат для управления обменом данными SSL.

Рассмотрим два базовых понятия протокола SSL, использующихся для описания его работы – сеанс SSL и соединение SSL.

Соединение (connection) – транспорт (в терминах OSI), обеспечивающий сервис некоторого подходящего типа. В случае SSL такие соединения представляют равноправные отношения между узлами. Соединения являются временными, и каждое соединение ассоциируется только с одним сеансом.

Сеанс (session) – это связь между клиентом и сервером, которая создается протоколом квитирования SSL. Сеанс определяет набор параметров криптографической защиты, которые могут использоваться несколькими соединениями. Сеансы

позволяют избежать необходимости ведения переговоров об установлении параметров защиты для каждого нового соединения.

Между любой парой обменивающихся информацией сторон можно установить несколько защищенных соединений. Допускается также установление нескольких одновременно существующих сеансов, но на практике такая возможность не используется.

Каждый сеанс характеризуется тем или иным состоянием. Состояние сеанса определяется следующими параметрами:

–идентификатор сеанса (session identifier) – произвольная последовательность байтов, генерируемая сервером для идентификации активного или возобновляемого состояния сеанса;

–сертификат узла (pear certificate) – сертификат X.509 стороны, участвующей в сеансе. Этот элемент состояния может быть пустым;

–метод сжатия (compression method) – алгоритм сжатия данных перед их шифрованием;

–параметры шифрования (cipher spec) – задают алгоритм шифрования данных и алгоритм хеширования, служит для вычисления значений MAC, определяют криптографические атрибуты;

–главный ключ (master secret) – 48-байтовый секретный ключ, используемый клиентом и сервером;

–флаг возобновления (is resumable) – признак, разрешающий использовать данный сеанс повторно для инициализации новых соединений.

Состояние соединения определяется следующими параметрами:

–случайный идентификатор – последовательность байтов, генерируемая для идентификации каждого соединения сервером и клиентом;

–ключ записи MAC сервера – секретный ключ, применяемый сервером при вычислении значений MAC для отправляемых им данных;

–ключ записи MAC клиента – секретный ключ, с помощью которого клиент вычисляет значения MAC для отправляемых им данных;

–ключ записи сервера – ключ традиционной схемы шифрования, используемый сервером для шифрования данных и клиентом для дешифрования этих данных;

–ключ записи клиента – ключ традиционной схемы шифрования, используемый клиентом для шифрования данных и сервером для дешифрования этих данных;

–векторы инициализации – значения, требуемые, например, при блочном шифровании в режиме сцепления шифрованных блоков (CBC) для каждого ключа. Первый раз это поле инициализируется протоколом квитирования SSL;

–порядковые номера – используются каждой из сторон для нумерации отправляемых и получаемых по каждому соединению сообщений. В случае получения или отправки какой-либо из сторон сообщения об изменении параметров существования, соответствующий порядковый номер устанавливается равным нулю. Значения порядковых номеров не должны превышать 2^{64}-1.

Протокол записи SSL обеспечивает поддержку сервисов конфиденциальности и целостности для соединений SSL. Протокол квитирования SSL определяет общий для клиента и сервера секретный ключ, используемый алгоритмом традиционной схемы для шифрования данных, передаваемых по протоколу SSL. Кроме того, протокол квитирования SSL определяет общий секретный ключ для вычисления значений MAC

Общая схема работы протокола записи SSL показана на рисунке 6.3.

Первым шагом является фрагментация. Сообщение, полученное от приложения более высокого уровня, разделяется на блоки размером не более 16384 байтов. Затем применяется сжатие, которое должно происходить без потерь и не должно увеличивать размер блока более чем на 1024 байта. По умолчанию в протоколах SSLv3 и TLS сжатие не применяется.

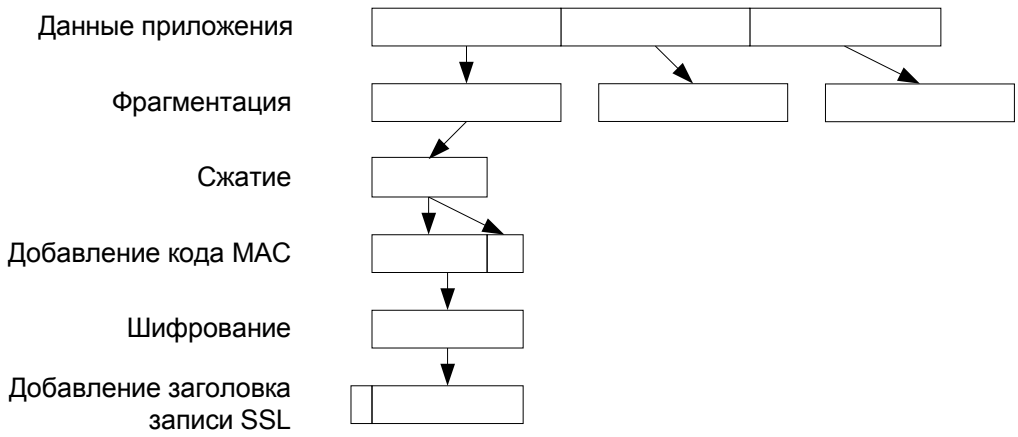

Рисунок 6.3 – Схема работы протокола записи SSL

Следующим шагом является вычисление кода аутентичности сообщения (значения MAC) с помощью общего секретного ключа. В качестве алгоритмов хеширования используются MD5 и SHA-1.

Затем сжатое сообщение с добавленным к нему кодом MAC шифруется с использованием симметричной схемы шифрования. Шифрование также не должно увеличивать длину блока более чем на 1024 байта, т.е. общий размер блока не может превысить 2^{14} + 2048 байтов. Допускается применение следующих алгоритмов шифрования:

–блочное шифрование:

–IDEA с длиной ключа 128 битов;

–RC2-40 с длиной ключа 40 битов;

–DES-40 с длиной ключа 40 битов;

–DES с длиной ключа 56 битов;

–3DES с длиной ключа 168 битов;

–Fortezza с 80-битовым ключом;

–поточное шифрование;

–RC4-40 с 40-битовым ключом;

–RC4-128 с длиной ключа 128 битов.

В случае применения алгоритмов поточного шифрования шифруются только сжатое сообщение и добавленное к нему значение MAC, а при использовании блочных шифров может потребоваться добавление заполнителя.

Завершающим шагом в работе протокола записи SSL является создание заголовка, состоящего из следующих полей:

–тип содержимого (8 битов) – определяет протокол верхнего уровня, с помощью которого должен обрабатываться данный фрагмент;

–старший номер версии (8 битов) – указывает старший номер версии используемого протокола (для SSLv3 поле содержит значение три);

–младший номер версии (8 битов) – указывает младший номер версии применяемого протокола SSL;

–длина сжатого фрагмента (16 битов) – длина данного фрагмента открытого текста (или сжатого фрагмента при сжатии) в байтах.

Для типа содержимого определены значения change_cipher_spec (для протокола изменения параметров шифрования), alert (для протокола извещения), handshake (для протокола квитирования) и application_data (для всех других протоколов верхнего уровня).

Протокол изменения параметров шифрования является самым простым из всех протоколов верхнего уровня стека протоколов SSL. Он генерирует однобайтовое сообщение, служащее для указания начать копирование параметров состояния ожидания в текущее состояние, что приводит к обновлению набора шифров, используемых для данного соединения.

Протокол извещения SSL предназначен для передачи другой участвующей в обмене стороны извещений, касающихся работы SSL. Как и данные любого другого приложения, использующего SSL, сообщения протокола извещения также сжимаются и шифруются в соответствии с параметрами текущего состояния.

Любое сообщение, генерируемое данным протоколом, состоит из двух байтов. Первый байт содержит значение, обозначающее уровень предупреждения или уровень неустранимой ошибки. Если указан уровень неустранимой ошибки, протокол SSL немедленно разрывает соединение. Другие соединения могут продолжать существовать, однако установление нового соединения для данного сеанса невозможно. Второй байт содержит код, обозначающий конкретный смысл сообщения. Извещения, указывающие на неустранимую ошибку:

–unexpected message – получено непригодное для обработки сообщение;

–bad_record_mac – получено неверное значение MAC;

–decompression_failure – функция распаковки получила неправильные входные данные (генерируется также в случае, если длина полученных в результате распаковки данных оказывается больше максимально допустимой);

–handshake_failure – отправитель не смог согласовать с получателем параметры защиты;

–illegal_parameter – значение поля в сообщении квитирования выходит за рамки допустимого диапазона или не согласуется со значениями других полей.

Кроме того, имеются следующие извещения:

–close_notify – извещает получателя о том, что отправитель больше не будет использовать данное соединение;

–no_certificate – может отправляться в ответ на запрос о наличии сертификата, если у отвечающей стороны соответствующего сертификата нет;

–bad_certificate – полученный сертификат поврежден (или не удалось выполнить верификацию подписи);

–unsupported_certificate – тип полученного сертификата не поддерживается;

–certificate_revoked – сертификат был отозван подписавшим его объектом;

–certificate_expired – срок действия сертификата истек;

–certificate_unknown – при обработке сертификата возникли какие-либо проблемы, вследствие чего его использование оказывается невозможным.

Протокол квитирования SSL позволяет серверу и клиенту выполнить взаимную аутентификацию, а также согласовать алгоритмы шифрования, вычисления MAC и криптографические ключи, которые будут служить для защиты передаваемых данных. Протокол квитирования используется до начала пересылки данных прикладных программ.

Все сообщения протокола квитирования имеют следующий формат:

–тип – указывает один из 10 допустимых типов сообщения, перечисленных в таблице 6.3;

–длина – длина сообщения в байтах;

–содержимое – параметры, связываемые с сообщением данного типа.

Таблица 6.3 – Типы сообщения протокола квитирования

Тип сообщения	Параметры
hello_request	не имеет параметров
client_hello	версия, случайные значения, идентификатор сеанса, комплект шифров, метод сжатия
server_hello	версия, случайные значения, идентификатор сеанса, комплект шифров, метод сжатия
certificate	цепочка сертификатов X.509
server_key_exchange	параметры обмена, подпись
certificate_request	тип сертификата, названия уполномоченных объектов
server_done	не имеет параметров
certificate_verify	подпись
client_key_exchange	параметры обмена, подпись
finished	хеш-значение

При установлении соединения между клиентом и сервером происходит обмен сообщениями, который можно условно разделить на четыре этапа (рисунок 6.4).

Первый этап – определение характеристик защиты. На этом этапе производится инициализация логического соединения и определяются связанные с ним характеристики защиты. Этот процесс инициализируется клиентом, который отправляет серверу сообщение client_hello со следующими параметрами:

–версия – наивысший номер версии SSL, поддерживаемый клиентом;

–случайное значение – генерируемое клиентом случайная структура, содержащая 32-битовую метку времени и 28 байтов, полученных с помощью генератора случайных чисел. Эти значения используются во время обмена ключами для защиты от атак воспроизведения;

–идентификатор сеанса – идентификатор переменной длины для данного сеанса. Ненулевое значение говорит о намерении клиента обновить параметры имеющегося соединения или создать новое соединение в рамках того же сеанса. Нулевое значение используется тогда, когда необходимо создать новое соединение в новом сеансе;

–комплект шифров – список, содержащий наборы криптографических алгоритмов, поддерживаемых клиентом, перечисленные в порядке убывания их приоритета. Каждый элемент списка определяет алгоритм обмена ключами и спецификации шифра (cipher spec);

–метод сжатия – список методов сжатия, поддерживаемых клиентом.

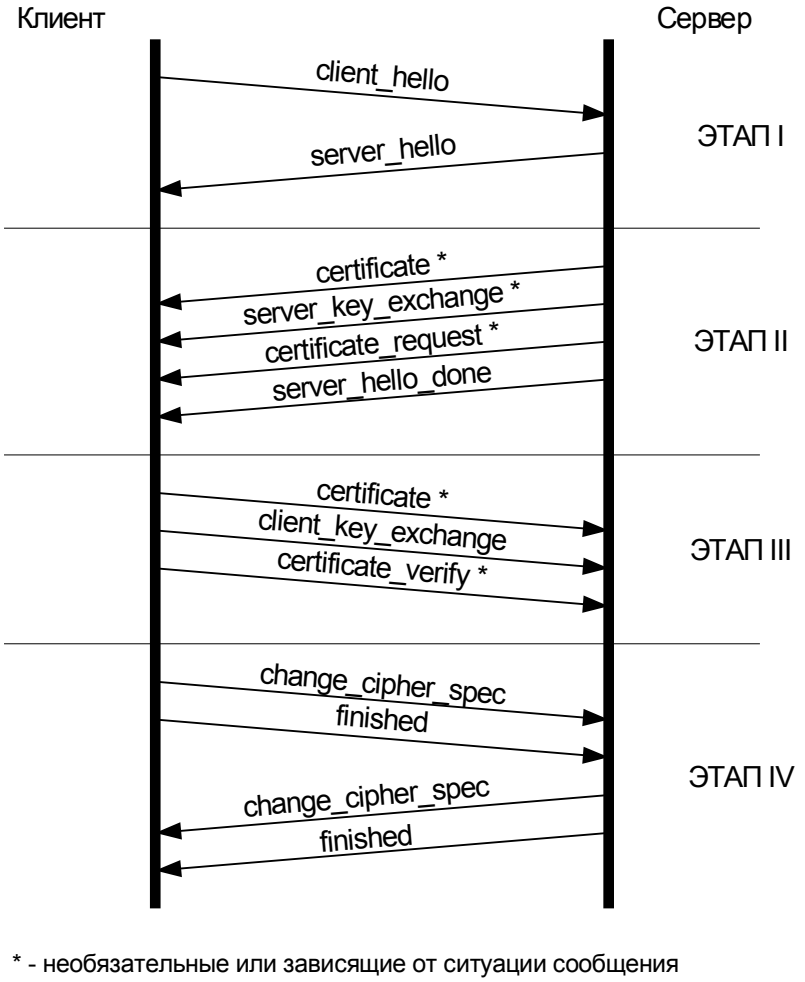

Рисунок 6.4 – Схема работы протокола квитирования SSL

После отправки сообщения client_hello клиент ожидает от сервера сообщения server_hello, которое имеет тот же формат, что

и client_hello. Поле Version (версия) содержит номер низшей из версий, поддерживаемых клиентом и высшей из версий, поддерживаемых сервером. Поле Random (случайное значение) генерируется сервером и не связано с соответствующим полем в сообщении клиента. Если значение поля SessionID (идентификатор сеанса) в сообщении клиента было отличным от нуля, то же значение применяется и сервером. В случае нулевого значения поле SessionID содержит значение идентификатора для нового сеанса. В поле CipherSuite (комплект шифров) определяется один комплект шифров, выбранных сервером из списка, предложенного клиентом. В поле Compression определяется метод сжатия, выбранный сервером из списка, предложенного клиентом.

Первым элементом поля CipherSuite определяется метод обмена ключами. Протокол SSL поддерживает следующие методы обмена ключами:

–RSA – секретный ключ шифруется с помощью открытого ключа RSA получателя. Для этого отправителю должен быть доступен сертификат открытого ключа получателя;

–метод Диффи-Хеллмана с фиксированными параметрами (Fixed Diffie-Hellman) – метод обмена ключами по Диффи-Хеллману, при котором сертификат сервера содержит открытые параметры алгоритма Диффи-Хеллмана, подписанные центром сертификации (Certificate Authority, CA). Клиент сообщает свои параметры открытого ключа алгоритма Диффи-Хеллмана либо в сертификате, если требуется аутентификация клиента, либо в сообщении обмена ключами;

–метод Диффи-Хеллмана с одноразовыми параметрами (Ephemeral Diffie-Hellman) – метод, служащий для создания временных одноразовых секретных ключей. В этом случае стороны обмениваются открытыми ключами Диффи-Хеллмана, подписанными с помощью личных ключей (RSA или DSS) отправителя. Для проверки подписи получатель может использовать соответствующий открытый ключ. Для аутентификации открытых ключей применяются сертификаты. Этот метод является самым безопасным, так как используется временный и подлежащий аутентификации ключ;

–анонимный метод Диффи-Хеллмана (Anonymous Diffie-Hellman) – использует базовый алгоритм Диффи-Хеллмана, но аутентификация не выполняется;

–Fortezza – метод, используемый в алгоритме Fortezza.

Затем следует элемент CipherSpec (параметры шифрования), состоящий из следующих полей:

–CipherAlgoritm – алгоритм шифрования (RC4, RC2, DES, 3DES, DES40, IDEA, Fortezza);

–MACAlgoritm – алгоритм вычисления MAC (MD5, SHA-1);

–CipherType – тип шифра: поточный (stream) или блочный (block);

–IsExportable – признак экспортируемости (true или false);

–HashSize – длина хеш-кода (0, 16 байтов для MD5, 20 байтов для SHA-1);

–KeyMaterial – параметры вычисления ключей (последовательность байтов, содержащих данные, используемые при генерировании ключей записи);

–IVSize – длина вектора инициализации для шифрования в режиме сцепления шифрованных блоков.

Второй этап – аутентификация и обмен ключами сервера. Данный этап начинается с отправки сервером его сертификата, если требуется аутентификация сервера. Отправляемое сообщение содержит сертификат X.509 или цепочку таких сертификатов. Сообщение certificate требуется для любого из предлагаемых методов обмена ключами, кроме анонимного метода Диффи-Хеллмана.

Затем при необходимости может быть отправлено сообщение server_key_exchange (обмен ключами сервера). Отправка такого сообщения не требуется в двух случаях: когда сервер отправил сертификат для метода Диффи-Хеллмана с фиксированными параметрами или когда используется схема обмена ключами RSA. Сообщение server_key_exchange необходимо при использовании следующих схем:

–анонимный метод Диффи-Хеллмана – сообщение содержит два глобальных значения, необходимых для использования метода Диффи-Хеллмана, а также открытый ключ сервера для алгоритма Диффи-Хеллмана;

–метод Диффи-Хеллмана с одноразовыми параметрами – сообщение содержит те же три параметра, что и в случае анонимного метода Диффи-Хеллмана, а также подпись этих параметров;

–обмен ключами по схеме RSA, когда использующий RSA сервер имеет ключ RSA только для подписи – в этом случае сервер должен создать временную пару ключей RSA и использовать сообщение server_key_exchange, чтобы передать клиенту открытый ключ из этой пары. Сообщение включает параметры временного открытого ключа RSA и подпись этих параметров;

–использование Fortezza.

После этого неанонимный (не использующий анонимный метод Диффи-Хеллмана) сервер может запросить сертификат клиента. Сообщение certificate_request (запрос сертификата) включает два параметра: certificate_type (тип сертификата) и certificate_authorities (центры сертификации). Тип сертификата указывает применяемый алгоритм шифрования с открытым ключом и может иметь следующие значения:

–RSA, только для подписи;

–DSS, только для подписи;

–RSA для метода Диффи-Хеллмана с фиксированными параметрами;

–DSS для метода Диффи-Хеллмана с фиксированными параметрами;

–RSA для метода Диффи-Хеллмана с одноразовыми параметрами;

–DSS для метода Диффи-Хеллмана с одноразовыми параметрами;

–Fortezza.

Вторым параметром сообщения certificate_request является список имен допустимых центров сертификации.

Последним сообщением второго этапа является сообщение server_done (готовность сервера), которым сервер извещает о завершении фазы приветствия ввиду отправки им всех соответствующих сообщений. После отправки этого сообщения сервер переходит в состояние ожидания сообщений от клиента.

Третий этап – аутентификация и обмен ключами клиента. Получив сообщение server_done, клиент должен убедиться в том, что сервер предоставил действительный сертификат (если это требуется) и что параметры сообщения server_hello являются приемлемыми. Если проверка завершается успешно, клиент отправляет серверу одно или несколько сообщений, рассмотренных ниже.

Если сервер запросил сертификат, клиент начинает данный этап с отправки серверу сообщения certificate. Если клиент не имеет подходящего сертификата, то он отправляет уведомление no_certificate.

Следующим отправляется сообщение client_key_exchange (обмен ключами клиента), которое является обязательным. Содержимое этого сообщения зависит от выбранного метода обмена ключами и может быть следующим:

–RSA – клиент генерирует 48-байтовый предварительный главный ключ и шифрует его с помощью открытого ключа из сертификата сервера или с помощью временного ключа RSA из сообщения server_key_exchange. Этот предварительный ключ используется для вычисления главного ключа;

–метод Диффи-Хеллмана с одноразовыми параметрами, анонимный метод Диффи-Хеллмана – отправляются открытые параметры клиента для метода Диффи-Хеллмана;

–метод Диффи-Хеллмана с фиксированными параметрами – содержимое сообщения является пустым, поскольку открытые параметры клиента уже были отправлены в сообщении certificate;

–Fortezza – отправляются параметры клиента для алгоритма Fortezza.

Далее клиент может отправить сообщение certificate-verify (проверка сертификата), чтобы обеспечить средства прямой верификации сертификата клиента. Это сообщение отправляется вслед за сертификатом клиента, поддерживающим подпись. Сообщение включает подпись хеш-кода предыдущего сообщения.

Четвертый этап – завершение. Клиент отправляет сообщение change_cipher_spec (изменение параметров шифрования) и копирует параметры CipherSpec состояния ожидания в поле CipherSpec текущего состояния. В ответ клиент немедленно

отправляет сообщение finished, шифрованное новым алгоритмом с новыми ключами и секретными значениями. Сообщение finished подтверждает, что процессы обмена ключами и аутентификации завершились успешно. Содержимое сообщения finished представляет собой значение хеш-кода всех сообщений квитирования, за исключением этого сообщения.

В ответ на эти два сообщения сервер посылает свое сообщение change_cipher_spec, копирует параметры CipherSpec состояния ожидания в текущее состояние и посылает свое сообщение finished.

На этом процесс квитирования завершается, и клиент и сервер могут начать обмен данными на уровне приложения.

Отметим, что существуют устройства, поддерживающие протокол SSL на аппаратном уровне или ускоряющие работу с ним. Протокол SSL как средство защиты данных в сети Internet получил широкое распространение. В первую очередь это связано с тем, что для работы с системой, использующей SSL, пользователям нет необходимости осуществлять какие-либо настройки своего компьютера и установленного на нем программного обеспечения.

6.3 Защита данных на пользовательском уровне модели OSI

Для обеспечения защиты данных в локальных сетях достаточно эффективным является применение различных фильтров пакетов и межсетевых экранов, однако данный подход оказывается неэффективным для обеспечения безопасной работы web-приложений. Рассмотрим подобную схему защиты для web-приложения, имеющего трехуровневую архитектуру (рисунок 6.5) [5].

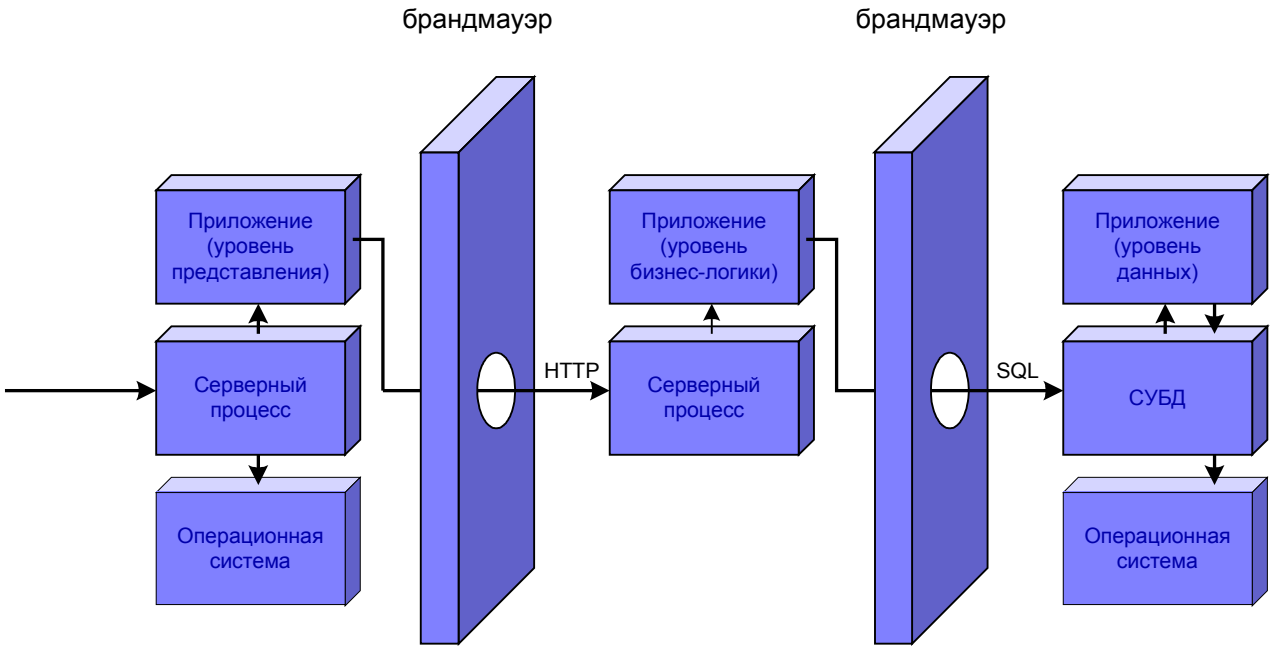

Рисунок 6.5 – Традиционная схема защиты сети с использованием брандмауэров

В данном случае каждый из уровней web-приложения – представления, логики и данных – реализован в виде отдельных компонентов, расположенных в разных зонах, защищаемых брандмауэрами.

Для работы такого web-приложения необходимо, чтобы первый брандмауэр обеспечивал прохождение HTTP-запросов, а второй – запросов к СУБД. Таким образом, компоненты web-приложения оказываются незащищенными от атак на пользовательском уровне OSI, так как брандмауэры не осуществляют проверку проходящих через них HTTP- или SQL-запросов.

Классическим приемом обеспечения безопасности web-приложений является надежное их программирование. При этом разработчики должны исходить из того, что злоумышленник в любой момент времени может манипулировать любыми значениями, такими как поля ввода, поля HTTP-запроса, информация о сеансе в URL, значения маркеров Cookie. Однако зачастую все вышеперечисленные требования не учитываются, что связано с двумя факторами: отсутствием знаний о технологии атак

на уровне приложения и сложностью и большими затратами на надежное программирование.

Альтернативой данному методу является использование брандмауэров для web-приложений (Web Application Firewall, WAF). Эти системы располагаются в сети перед web-приложениями и обеспечивают более широкие возможности по сравнению с обычными брандмауэрами. В отличие от классических брандмауэров, WAF интерпретируют защищаемые web-приложения с их отдельными web-страницами, сценариями, полями ввода, значениями параметров и маркерами Cookie.

Таким образом, WAF в состоянии контролировать объекты в вызываемом URL и параметры GET и POST запросов, а также блокировать вредоносные запросы. Кроме того, WAF позволяет обеспечить возможность вызова только тех URL, которые принадлежат приложению, и блокировать доступ ко всем остальным ресурсам сервера.

Еще одной возможностью WAF является контроль откликов и их блокирование в случае необходимости. Отклик web-сервера в стандартной конфигурации содержит информацию о самом web-сервере и его конфигурации, а также об используемой операционной системе (например, Apache 1.3.34 (Linux), mod_ssl 2.8.25).

Обычно WAF используют два подхода к управлению безопасностью – в соответствии с положительной или отрицательной логикой безопасности. В первом случае действием по умолчанию является запрещение, и определяются разрешенные действия, во втором – определяются только запрещенные действия и разрешаются все остальные. WAF используют заранее составленные «черные списки», содержащие схемы распространенных типов атак. Тем самым обеспечивается защита от многих известных атак, базирующихся на этих схемах, и нет необходимости в конфигурировании всех параметров WAF. WAF должен допускать исключение из области проверки определенных полей, например, содержащих коды HTML, используемых для работы web-приложения.

Положительная логика, также называемая «белым списком», предполагает определение разрешенных действий, и является более

консервативной. Положительная логика требует достаточно больших затрат на конфигурацию, однако в результате обеспечивается более надежная защита, в том числе и от известных типов атак.

Наиболее эффективным является WAF, который может соответствующим образом интерпретировать web-приложение, т.е. в этом случае преобладает положительная логика. Основой для этого являются специально составленные правила, называемые политиками. Возможны два подхода к формированию политик: динамический и статический.

При динамическом подходе действительные страницы входа и работа WAF определяются в реальном времени. Для каждой запрошенной пользователем страницы содержащиеся в коде HTML ссылки, включая определенные величины, включаются в динамически создаваемое правило. Однако данный подход неприменим при использовании на стороне клиента различных сценариев (например, написанных на JavaScript), так как они позволяют генерировать значения и адреса. Вследствие этого необходимо определять набор соответствующих исключений.

Статический подход является более распространенным. При этом правила создаются до использования WAF вручную или с использованием специальных средств. Например, можно прозрачно включить WAF в поток данных web-приложения для предварительного протоколирования трафика. На основе полученных данных формируются соответствующие правила безопасности.

В некоторых случаях WAF предоставляют также возможность определения пути навигации, по которому может быть вызвана данная web-страница. Кроме того, WAF обеспечивает проверку статических и динамических значений, используемых сценариями, таких как идентификаторы сеанса. WAF обеспечивает также проверку маркеров Cookie и предотвращает попадание в web-приложение модифицированных пользователем маркеров.

Практически все WAF поддерживают работу с протоколом SSL и аппаратное ускорение шифрования и дешифрования, так как в противном случае был бы возможен только анализ незашифрованных данных.

7 Лабораторный практикум к курсу

7.1 Лабораторная работа № 1. Основы теории чисел

7.1.1 Постановка задачи

Цель работы: изучить основные понятия, теоремы и алгоритмы теории чисел, используемые в криптографии.

Задание: Составить схему алгоритма и написать программу, реализующую следующие функции:

–нахождения наибольшего общего делителя двух чисел на основе алгоритма Евклида;

–нахождения последовательности простых чисел, не превосходящих данного N, на основе алгоритма Эратосфена;

–отыскания функции Эйлера для положительного целого числа;

–отыскания решения линейного сравнения.

Полученные алгоритмы оформить в виде отдельного модуля (библиотеки).

7.1.2 Теоретические предпосылки

Основа любого криптографического алгоритма базируется на элементах теории чисел, изучению которых и посвящена данная лабораторная работа. Теория чисел занимается изучением свойств целых чисел. Сразу оговоримся, что при изложении теоретического материала мы будем обозначать буквами только целые числа. [4, 25]

1. Основные понятия.

Определение 1. Если a делится на b нацело, мы будем говорить, что b делит a. При этом a называется кратным числа b, b – делителем числа a. Число a можно представить как

$$a = q \cdot b,$$

где q – полное частное.

Теорема 1 Если в равенстве вида $k + l + ... + n = p + q + ... + s$ относительно всех членов, кроме какого-либо одного, известно, что они кратны b, то и этот один член кратен b.

Доказательство:

Пусть таким одним членом будет k. Имеем

$$l = b \cdot l_1$$

$$n = b \cdot n_1$$

$$p = b \cdot p_1$$

$$q = b \cdot q_1$$

$$s = b \cdot s_1$$

Перенесем в правую часть равенства все члены, кроме k.

$$k = p + q + ... + s - l - ... - n = b \cdot (p_1 + q_1 + ... + s_1 - l_1 - ... - n_1)$$

Таким образом, k представляется произведением b на целое число $(p_1 + q_1 + ... + s_1 - l_1 - ... - n_1)$ и тем самым делится на b по определению 1. Что и требовалось доказать.

Теорема 2 (о делении с остатком).

Всякое целое a представляется единственным способом с помощью положительного целого b равенством вида

$$a = b \cdot q + r; \qquad 0 \le r \prec b \tag{7.1}$$

Доказательство

Действительно, одно представление числа a равенством вида (7.1) получим, взяв $b \cdot q$ равным наибольшему кратному числа b, не превосходящему a.

Допустим существование другого представления числа a еще одним равенством вида (7.1)

$$a = b \cdot q_1 + r_1; \qquad 0 \le r_1 \prec b \tag{7.2}$$

Вычтем почленно равенство (7.2) из (7.1), получим

$$0 = b \cdot (q - q_1) + r - r_1 \qquad (7.3)$$

Согласно Теореме 1 разность $r - r_1$ кратна b. С другой стороны разность двух неотрицательных чисел меньших b сама будет численно меньше b. Числом кратным b и численно меньшим b является 0. Если $r - r_1 = 0$, то из равенства (7.3) следует, что и $q - q_1 = 0$. Таким образом, второе представление числа a тождественно первому. Теорема 2 доказана.

Число q называется *неполным частным,* а число r – *остатком от деления* a на b.

2. Наибольший общий делитель.

Определение 2.Число, которое делит каждое из чисел a и b, называется общим делителем чисел a и b.

Определение 3. Наибольший из делителей чисел a и b называется наибольшим общим делителем (НОД) этой пары чисел.

Обозначение. НОД (a,b) ≡ (a,b).

Определение 4. Если НОД(a,b)=1, то числа a и b называются попарно простыми.

Теорема 3. Если a кратно b, то совокупность общих делителей чисел a и b совпадает с совокупностью делителей одного числа b; в частности (a,b)=b.

Доказательство:

Действительно, всякий общий делитель чисел a и b является делителем и одного b. Раз a кратно b, то, согласно Теореме 1, всякий делитель числа b является также делителем числа a, т.е. является общим делителем чисел a и b.

Таким образом, совокупность общих делителей чисел a и b совпадает с совокупностью делителей одного b. Поскольку наибольший делитель числа b есть само b, то (a,b)=b.

Теорема 4. Если $a = b \cdot q + r$, то совокупность общих делителей чисел a и b совпадает с совокупностью делителей чисел b и c; в частности (a,b)=(b,c).

Доказательство:

Действительно, всякий общий делитель чисел a и b согласно Теореме 3 является и общим делителем чисел b и c. Всякий общий делитель чисел b и c делит a и, следовательно, является общим делителем чисел a и b.

Таким образом, общие делители чисел a и b те же, что и общие делители чисел b и c. В частности, должны совпадать и наибольшие из этих делителей, (a,b)=(b,c).

Алгоритм Евклида. Применяется для отыскания НОД двух чисел.

Пусть a,b – положительные целые и a>b.

Согласно Теореме 3 находим ряд равенств:

$$\begin{aligned}
&a = b \cdot q_1 + r_2, && 0 \le r_2 \prec b,\\
&b = r_2 \cdot q_2 + r_3, && 0 \le r_3 \prec r_2,\\
&r_2 = r_3 \cdot q_3 + r_4, && 0 \le r_4 \prec r_3,\\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\
&r_{n-2} = r_{n-1} \cdot q_{n-1} + r_n, && 0 \le r_n \prec r_{n-1},\\
&r_{n-1} = r_n \cdot q_n,
\end{aligned} \qquad (7.4)$$

заканчивающийся, когда получается некоторое $r_{n+1}=0$. Последнее неизбежно, так как ряд $b, r_2, r_3, \ldots$ не может содержать более чем b положительных чисел.

Таким образом, НОД чисел a и b равен последнему не равному нулю остатку алгоритма Евклида. Поскольку, согласно Теореме 4

$(a,b) = (b,r_2) = (r_2,r_3) = \ldots = (r_{n-1},r_n) = r_n$.

Пример. Отыскать НОД(25,9) - ?

$25 = 9*2 + 7$

$9 = 7*1 + 2$

$7 = 2*3 + 1$

$2 = 1*2 + 0$

НОД(25,9) = 1

3. Простые числа.

Определение 5. Всякое a>1 будем называть простым, если у него нет других делителей, кроме 1 и самого себя, иначе – составным.

Определение 6. Два числа называются попарно простыми, если их НОД равен 1.

Алгоритм Эратосфена построения последовательности простых чисел в ряду целых чисел, не превосходящих данного целого N.

Выписываем ряд чисел

$$1,2,\ldots, N \qquad (7.5)$$

Первое простое число в ряду (7.5) – 2. Вычеркиваем из ряда (7.5) все числа кратные 2, кроме самого числа 2. Первое, оставшееся после 2, простое число – 3. Вычеркиваем из ряда (7.5) все числа кратные 3, кроме самого числа 3. Первое, следующее за 3, невычеркнутое простое число 5. Вычеркиваем из ряда (7.5) все числа кратные 5, кроме числа 5. И т.д.

Когда указанным способом вычеркнуты все числа, кратные простых, меньше простого p, то все невычеркнутые меньшие p^2 будут простые.

Выводы (без доказательства):

1) приступая к вычеркиванию кратных простого p, это вычеркивание следует начать с p^2;

2) составление последовательности простых чисел, не превосходящих N, закончено, как только вычеркнуты все составные кратные простых, не превосходящих $\sqrt{N}$.

Теорема 5. Всякое целое, большее единицы, разлагается на произведение простых сомножителей и притом единственным способом:

$$a = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_K^{\alpha_K}, \qquad (7.6)$$

где $p_1,\ldots,p_K$ – простые сомножители числа a,

$\alpha_1,\ldots,\alpha_K$ – кратности вхождения соответственно сомножителей $p_1,\ldots,p_K$ в число a.

Разложение (7.6) числа a на простые сомножители называется *каноническим.*

Пример. $18 = 2^1 \cdot 3^2$

Определение 7. Функцией Эйлера $\varphi(a)$ называется функция, которая для $\forall a \in Z_+$, равна количеству чисел в ряду от 1 до a-1 попарно простых с a, где $a \geq 1$.

Пример.

$\varphi(1) = 1$ $\varphi(4) = 2$ $(1,3)$

$\varphi(2) = 1$ $\varphi(5) = 4$ $(1,2,3,4)$

$\varphi(3) = 2$ $\varphi(6) = 2$ $(1,5)$ и тт.д

Теорема 6. Пусть $a = p_1^{\alpha_1} \cdot p_2^{\alpha_2} \cdot \ldots \cdot p_k^{\alpha_k}$ - каноническое разложение числа a. Тогда имеем

$$\varphi(a) = a \cdot \left(1 - \frac{1}{p_1}\right) \cdot \left(1 - \frac{1}{p_2}\right) \cdot \ldots \cdot \left(1 - \frac{1}{p_k}\right)$$

или также

$$\varphi(a) = \left(p_1^{\alpha_1} - p_1^{\alpha_1 - 1}\right) \cdot \left(p_2^{\alpha_2} - p_2^{\alpha_2 - 1}\right) \cdot \ldots \cdot \left(p_k^{\alpha_k} - p_k^{\alpha_k - 1}\right).$$

(без доказательства).

В частности, если p – простое, то $\varphi(p) = p - 1$.

4. Вычеты. Линейные сравнения.

Определение 8. Пусть m – некоторое целое положительное число m>1. Пусть a и b – это числа, которые при делении на m имеют один и тот же остаток:

$a = m \cdot t_1 + r \qquad 0 \le r < m$

$b = m \cdot t_2 + r$

Числа a и b будем называть равноостаточными.

Очевидно, таких чисел бесчисленное множество, т.е. они образуют класс чисел равноостаточных по модулю m или, как говорят, сравнимых по модулю m.

Обозначение. $a = b \bmod m$.

Пример. $7 = 10 \bmod 3 \qquad 7 = 10 + 3 \cdot (-1)$.

Все сравнимые числа отличаются на кратное число модулей и поэтому могут быть представлены в виде $a = b + m \cdot t$.

Сравнение первой степени, или линейное сравнение

$$a \cdot x + b = 0 \bmod m \qquad (7.7)$$

можно представить в виде:

$$a \cdot x = b \bmod m \qquad (7.8)$$

перенесением свободного члена в правую часть.

Среди множества различных линейных сравнений, нас будут интересовать только те, для которых выполняется следующее условие (a,b)=1, поскольку в этом случае линейное сравнение будет иметь решение и притом единственное.

Рассмотрим алгоритм решения линейного сравнения, основанный на алгоритме Евклида.

1. С помощью алгоритма Евклида находят вектор неполных частных $q_1, \ldots, q_n$ для чисел m и a (порядок чисел в данном случае принципиален).

2. Делают обозначение: $P_0=1$, $P_1=q_1$.

3. По формуле

$$P_i = q_i \cdot P_{i-1} + P_{i-2}$$

находят значения вектора P_i для i=2..n-1.

4. Решение линейного сравнения находят по формуле

$$x = (-1)^{n-1} \cdot P_{n-1} \cdot b \bmod m$$

Пример. Решить линейное сравнение $7x = 5 \bmod 19$

Решение

1. Находим неполные частные с использованием алгоритма Евклида для чисел m и a.

$19 = 7 \cdot 2 + 5 \quad q_1 = 2$

$7 = 5 \cdot 1 + 2 \quad q_2 = 1$

$5 = 2 \cdot 2 + 1 \quad q_3 = 2$

$2 = 1 \cdot 2 \quad q_4 = 2$

n=4

2. $P_0=1$, $P_1=q_1$.

3.

$P_2 = q_2 \cdot P_1 + P_0 \qquad P_2 = 1 \cdot 2 + 1 = 3$

$P_3 = q_3 \cdot P_2 + P_1 \qquad P_3 = 2 \cdot 3 + 2 = 8$

4. $x = (-1)^3 \cdot P_3 \cdot b \bmod m = (-1) \cdot 8 \cdot 5 \bmod 19 = -40 \bmod 19 = -2 \bmod 19 = 17 \bmod 19$

7.1.3 Упражнения

1. Найти НОД(396,1452)

2. Найти каноническое разложение числа а)2156; б)1934; в)1132.

3. Являются ли числа попарно простыми

а) (678, 941); б) (243, 1485); в) (535, 321)?

4. Решить линейное сравнение:

а) $5x = 4 \bmod 21$; б) $11x = 17 \bmod 119$; в) $12x = 8 \bmod 67$

7.1.4 Вопросы и задания к лабораторной работе № 1

1. Докажите теорему о делении с остатком.
2. Дайте понятие НОД двух чисел. Какой алгоритм используется для его нахождения?
3. Какие числа называются простыми, составными?
4. Расскажите алгоритм нахождения последовательности простых чисел.
5. Какое разложение целого числа называется каноническим?
6. Что показывает функция Эйлера, как она рассчитывается?
7. Расскажите алгоритм решения линейных сравнений.

7.2 Лабораторная работа № 2. Криптографические системы

7.2.1 Постановка задачи

Цель работы: изучить основные криптографические алгоритмы.

Задание: Составить схему алгоритма и написать программу, реализующую:

–симметричные алгоритмы шифрования простой и сложной заменой;

–один из асимметричных алгоритмов шифрования (по выбору).

7.2.2 Теоретические предпосылки

Представлены в разделе 2.

7.2.3 Вопросы и задания к лабораторной работе № 2

1. Что изучает наука криптография?
2. Что понимается под шифром, ключом?
3. Расскажите общий принцип действия любой криптографической системы.
4. Особенности функционирования симметричных криптосистем. На чем основана криптостойкость симметричных алгоритмов шифрования?
5. Перечислите и кратко охарактеризуйте основные виды симметричных алгоритмов шифрования.
6. Расскажите суть шифров простой и сложной замены, их основные недостатки.
7. Какие современные симметричные криптосистемы Вы знаете?
8. Особенности функционирования асимметричных алгоритмов шифрования.
9. Дайте понятие односторонней функции. Какие односторонние функции используются в криптографических алгоритмах шифрования?
10. Расскажите основные асимметричные криптографические алгоритмы RSA и El Gamal, на чем основана их криптостойкость?

7.3 Лабораторная работа № 3. Электронная цифровая подпись

7.3.1 Постановка задачи

Составить схему алгоритма и написать программу, выполняющую следующие функции:

Серверная часть:

–формирование ЭЦП произвольного сообщения M на основе алгоритма; в качестве хэш-функции можно взять любой алгоритм, например, MD2;

–пересылка сообщения M и ЭЦП клиенту.

Входные данные: файл исходного сообщения M (произвольной длины).

Выходные данные: значение ЭЦП.

Клиентская часть: проверка ЭЦП.

Входные данные: сообщение M, значение ЭЦП.

Выходные данные: сообщение о том подтверждена или отклонена проверка целостности документа.

7.3.2 Теоретические предпосылки

Представлены в разделе 3.

7.3.3 Вопросы и задания к лабораторной работе № 3

1. Перечислите виды злоумышленных действий, защитой от которых служит ЭЦП. Приведите примеры
2. Перечислите основные отличия цифровой подписи от обычной.
3. Дайте определение хэш-функции. С какой целью используется хэш-функция при формировании и проверке ЭЦП?
4. Какой ключ (открытый или секретный) используется для формирования подписи, а какой для ее проверки?

7.4 Лабораторная работа № 4. Реализация алгоритма распределения ключей Диффи-Хеллмана

7.4.1 Постановка задачи

Составить схему алгоритма и написать программу, выполняющую расчет общего секретного ключа двух абонентов сети на основе алгоритма Диффи-Хеллмана. Программа должна одновременно запускаться для двух

абонентов сети одновременно и осуществлять пересылку по сети значений Ya и Yb между абонентами.

7.4.2 Теоретические предпосылки

Представлены в разделе 3.

7.4.3 Вопросы и задания к лабораторной работе № 4

1. Какие функции выполняет управление ключевой информацией?
2. Какие разновидности мехнизмов управления ключами выделяется?
3. Какая сложная математическая задача лежит в основе алгоритма Диффи-Хеллмана?
4. Назовите системные и прикладные программные продукты, использующие алгоритм Диффи-Хеллмана для распределения ключей.

7.5 Лабораторная работа № 5. Изучение политики безопасности операционной системы Windows 2000

7.5.1 Постановка задачи

1. Изучить структуру и назначение Active Directory и ее объектов.
2. Научиться управлять группами, компьютерами, учетными записями пользователей.
3. Изучить структуру и возможности групповых и локальных политик безопасности.
4. Научиться настраивать политику безопасности:

–политику учетных записей;

–глобальную и локальную политики безопасности;

–журнал событий и другие средства защиты и администрирования.

7.5.2 Теоретические предпосылки

Структура и назначение Active Directory и ее объектов.

Служба каталогов Active Directory является средством для именования, хранения и выборки информации в некоторой распределенной среде, доступное для приложений, пользователей и различных клиентов этой среды. Служба сетевых каталогов хранит информацию об общедоступных приложениях, файлах, принтерах и сведения о пользователях. [18]

Служба каталогов Active Directory обеспечивает эффективную работу сложной корпоративной среды, предоставляя следующие возможности.

Единая регистрация в сети. Пользователи могут регистрироваться в сети *с одним* именем и паролем и получать при этом доступ ко всем сетевым ресурсам (серверам, принтерам, приложениям, файлам и т. д.) независимо от их расположения в сети.

Безопасность информации. Средства аутентификации и управления доступом к ресурсам, встроенные в службу Active Directory, обеспечивают централизованную защиту сети. Права доступа можно определять не только для каждого *объекта* каталога, но и каждого *свойства* (атрибута) объекта.

Централизованное управление. Администраторы могут централизованно управлять всеми корпоративными ресурсами.

Администрирование с использованием групповых политик. При загрузке компьютера или регистрации пользователя в системе выполняются требования групповых политик; их настройки хранятся в объектах групповых политик (GPO) и "привязываются" к сайтам, доменам или организационным единицам. Групповые политики определяют, например, права доступа к различным объектам каталога или ресурсам, а также множество других правил работы в системе.

Интеграция с DNS. Служба Active Directory тесно связана с DNS. Этим достигается единство в именовании ресурсов локальной сети и сети Интернет, в результате чего упрощается подключение пользовательской сети к Интернету.

Масштабируемость. Служба Active Directory может охватывать как один домен, так и множество доменов, один контроллер домена или множество контроллеров домена, т. е. она отвечает требованиям сетей любого масштаба. Несколько доменов можно объединить в дерево доменов, а несколько деревьев доменов можно связать в лес.

Стандартные интерфейсы. Для разработчиков приложений служба каталогов предоставляет доступ ко всем возможностям (средствам) каталога и поддерживает принятые стандарты и интерфейсы программирования (API). Служба каталогов тесно связана с операционной системой, что позволяет избежать дублирования в прикладных программах функциональных возможностей системы, например, средств безопасности.

Рассмотрим основные понятия, используемые в Active Directory.

Каталог состоит из *элементов* (entries), представляющих собой информацию, или *атрибуты,* связанные с некоторым реальным *объектом,* например компьютером, человеком или организацией.

Каждый объект принадлежит хотя бы к одному *объектному классу,* представляющему собой некоторое семейство объектов с определенными общими характеристиками. Класс объектов определяет тип информации, содержащейся в Active Directory для экземпляров (объектов) данного класса. Атрибуты могут быть как *обязательными* (mandatory) для данного класса (например, имя), так и *дополнительными* (optional) (пароль).

Контейнер (container) - это специфический объект службы каталогов, который, в отличие от обычных объектов, не имеет какого-либо физического представления, а служит только структурной организации других объектов каталога. Типичным примером контейнеров могут служить *организационные единицы,*

или *подразделения,* используемые для упрощения администрирования отдельных групп ресурсов или пользователей в домене.

Элементы каталога организованы в виде *иерархического дерева,* называемого Directory Information Tree (DIT, Информационное дерево каталога или просто Дерево каталога). Элементы, находящиеся ближе к корню дерева, обычно представляют крупные объекты, например, организации или компании; элементы, располагающиеся на ветвях этого дерева (листья), представляют более простые объекты - пользователей, устройства, компьютеры.

Схема каталога (Directory Schema) - это набор правил, описывающих структуру дерева каталога, объявления и синтаксис объектных классов и типы атрибутов, входящих в каталог.

Схема каталога гарантирует, что все добавления или изменения каталога соответствуют данным правилам, и препятствует появлению некорректных элементов, ошибочных типов атрибутов или классов.

В Active Directory схема реализована как набор экземпляров объектных классов, хранящийся в самом каталоге. Этим Active Directory отличается от многих каталогов, в которых схема хранится в текстовом файле, считываемом при запуске каталога. Когда схема хранится в каталоге, пользовательские приложения могут обращаться к ней и узнавать об имеющихся объектах и свойствах. Схему Active Directory можно динамически обновлять: модифицировать и расширять.

Основные компоненты любой службы каталога - база данных, содержащая нужную информацию, и один или несколько протоколов, обеспечивающих доставку данных пользователям.

Active Directory обеспечивает хранение любой общедоступной информации. Как и другие службы каталогов, Active Directory обеспечивает некоторый механизм хранения информации и протоколы для доступа к ней.

Компьютеры на базе Windows 2000 объединяются в домены. *Домены* - это известное решение для администрирования групп, предоставляющее каждому пользователю учетную запись в конкретном домене. Однако, в отличие от Windows NT Server 4.0,

где доменам давались простые строковые имена (имена NetBIOS), в среде Windows 2000 Server каждый домен должен иметь имя, отвечающее соглашениям именования доменов Domain Name System (DNS). В каждом домене один или несколько компьютеров должны выполнять функции контроллеров домена.

В среде Windows 2000 Server каждый контроллер домена содержит полную копию базы данных Active Directory этого домена. В Active Directory используются так называемое ядро Extended Storage Engine (ESE) и два различных протокола, обеспечивающих связь между клиентами и базой данных. Для поиска контроллера домена клиент обращается к протоколу, описанному в DNS. Для доступа к данным в Active Directory клиент использует протокол Lightweight Directory Access Protocol (LDAP).

В большинстве современных сетей TCP/IP используется служба DNS, главное назначение которой преобразовывать символьные имена в IP-адреса. Для этого каждый компьютер-сервер DNS имеет набор записей с информацией о ресурсах. Каждая запись имеет некоторый тип, определяющий характер и назначение хранящейся информации. Интеграцию служб Active Directory и DNS можно рассматривать в трех аспектах:

–домены Active Directory и домены DNS имеют одинаковую иерархическую структуру и схожее пространство имен;

–зоны (zone) DNS могут храниться в Active Directory. Если используется сервер DNS, входящий в состав Windows 2000 Server, то первичные зоны (primary zone), занесенные в каталог, реплицируются на все контроллеры домена, что обеспечивает лучшую защищенность службы DNS.

Каждый элемент Active Directory и каждый атрибут любого элемента имеют список управления доступом (ACL), который определяет права и возможности пользователей в отношении доступа к конкретным элементам и атрибутам. Например, список ACL может позволить одним пользователям читать атрибуты некоторого элемента, другим пользователям – читать и изменять некоторые из атрибутов, а остальным – запретить какой-либо доступ к элементу. Эффективное управление доступом невозможно

без достоверной аутентификации клиентов, Active Directory использует для этой цели протокол Kerberos.

Управление подразделениями, компьютерами, группами и учетными записями пользователей.

Для управления учетными записями пользователей и компьютерами следует вначале войти в раздел администрирования (Administrative Tools) и выбрать Active Directory Users and Computers (рисунок 7.1).

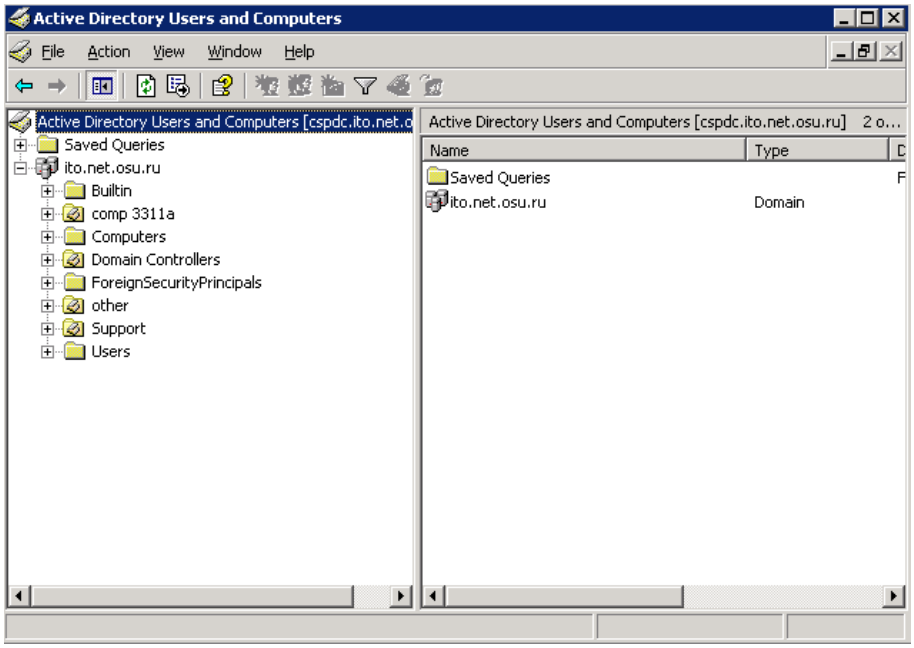

Рисунок 7.1

Создание подразделения (организационной единицы).

Для создания *подразделения,* или *организационной единицы* (Organizational Unit, OU) следует:

1. Выделить объект типа "домен" (на рисунке 1 домен имеет название ito.net.osu.ru) и нажмать правую кнопку мыши. В появившемся меню выберать команду Создать | Подразделение (New | Organizational Unit). Можно воспользоваться панелью инструментов и кнопкой Создание нового подразделения в текущем контейнере (Create a new organizational unit in a current container) на панели инструментов.

2. В открывшемся окне указать имя создаваемого подразделения и нажмите кнопку ОК.

В результате в выбранном вами домене будет создано подразделение с заданным именем. В дальнейшем внутри него можно создать вложенные подразделения.

На рисунке 7.1 представлено несколько подразделений – comp3311a, other, Support, Domain Controllers.

Создание группы.

В процессе установки домена Windows 2000 в нем создается несколько встроенных групп, обладающих определенным набором прав. Их можно использовать для присвоения администраторам или пользователям определенных ролей или прав доступа в домене.

К встроенным относятся перечисленные ниже группы. Эти группы служат для назначения разрешений доступа пользователям, на которых возложено выполнение в данном домене каких-либо административных функций.

Локальные группы в домене:

–Администраторы (Administrators);

–Гости (Guests);

–Операторы архива (Backup Operators);

–Операторы печати (Print Operators);

–Операторы сервера (Server Operators);

–Операторы учета (Account Operators);

–Пользователи (Users);

–Репликатор (Replicator).

–Совместимый с пред-Windows 2000 доступ (Pre-Windows 2000 CompatibleAccess).

Глобальные группы:

–Администраторы домена (Domain Admins);

–Владельцы-создатели групповой политики (Group Policy Creator Owners);

–Гости домена (Domain Guests);

–Издатели сертификатов (Cert Publishers);

–Компьютеры домена (Domain Computers);

–Контроллеры домена (Domain Controllers);

–Пользователи домена (Domain Users);

Универсальные группы:

–Администраторы предприятия (Enterprise Admins);

–Администраторы схемы (Schema Admins).

Универсальные группы создаются только на контроллерах корневого (первого в лесе) домена. В зависимости от установленных на сервере служб могут быть и дополнительные встроенные группы, локальные в домене или глобальные. По умолчанию все встроенные локальные группы домена находятся в папке Builtin объекта домена (рисунок 7.2). Все встроенные глобальные группы находятся в папке Users. Встроенные группы можно переносить в другие контейнеры или подразделения в пределах домена.

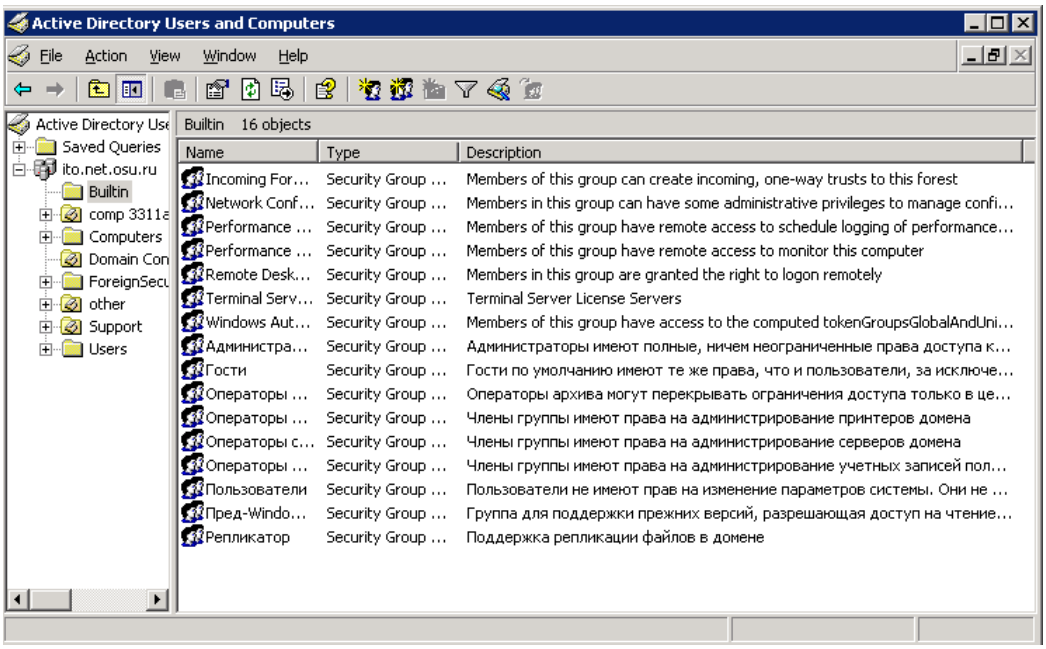

Рисунок 7.2

По умолчанию каждая созданная в домене учетная запись автоматически становится членом группы Пользователи домена. Кроме того, группа Пользователи домена является членом локальной в домене группы Пользователи.

Любой объект типа Компьютер (Computer) при создании по умолчанию автоматически включается в группу Компьютеры домена.

Группа Администраторы домена объединяет всех пользователей, имеющих полный административный доступ в домене. По умолчанию Администраторы домена являются членами локальной в домене группы Администраторы.

Группа Гости домена объединяет все учетные записи, с помощью которых можно зарегистрироваться в домене без пароля и получить минимальные права доступа. По умолчанию Гости домена являются членами локальной в домене группы Гости.

Помимо перечисленных выше встроенных групп администратор может создать любое количество групп пользователей и предоставить им необходимый набор прав и разрешений. Для создания группы необходимо выполнить следующее:

1. Выбрать подразделение, где следует создать группу, и нажмите правую кнопку мыши. Выберать в появившемся меню команду Создать | Группа (Group), либо нажать кнопку Создание новой группы в текущем контейнере (Create New Group in a Current Container) на панели инструментов.

2. В открывшемся окне диалога Новый объект – Группа (New Object - Group) в поле Имя группы (Group name) ввести имя создаваемой группы.

3. Установить переключатель Тип группы (Group type) в одно из положений, соответствующее типу создаваемой группы: Группа безопасности (Security) или Группа распространения (Distribution). Первый тип группы служит для предоставления пользователям определенного набора прав доступа к таким ресурсам сети, как файлы и принтеры. Второй тип группы служит только для распространения информации в сети, например в качестве списков рассылки электронной почты. Следует отметить, что группы безопасности могут использоваться в качестве групп распространения.

4. Установив в одно из положений переключатель Область действия группы (Group scope), выберать подходящую область действия создаваемой группы. *Область действия группы* определяет, где может быть видна данная группа *(уровень доступности)* и какие типы объектов могут быть ее членами, и может быть выбрана как:

–локальная в домене (Domain Local): пользователи, а также глобальные и универсальные группы из всего леса, другие локальные группы из этого же домена;

–глобальная (Global): пользователи, а также глобальные и универсальные группы;

–универсальная (Universal): пользователи и глобальные группы (только в основном режиме домена).

Создание учетной записи пользователя.

Для создания в домене учетной записи пользователя, предположим с идентификатором popov_as, необходимо выполнить следующее:

1. Указать подразделение, в котором следует создать учетную запись, и нажмите правую кнопку мыши. В появившемся меню выберать команду Создать | Пользователь.

2. В окне диалога Новый объект - Пользователь (New Object - User) в поле Имя входа пользователя (User logon name) ввести уникальный идентификатор, в поле Имя (First name) - имя пользователя, в поле Фамилия (Last name) - фамилию пользователя, в поле Полное имя (Full name) автоматически появятся имя и фамилия пользователя (рисунок 7.3). После ввода всей необходимой информации нажать кнопку Далее (Next).

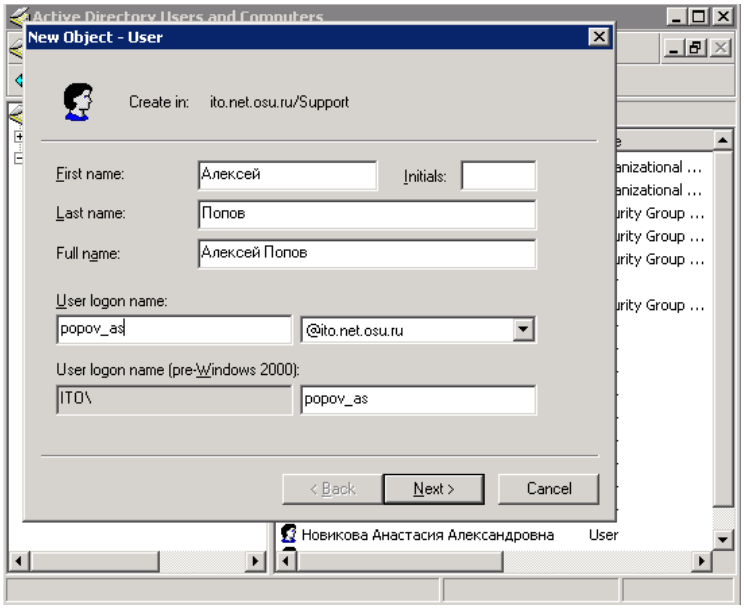

Рисунок 7.3

3. В следующем окне в полях ввода Пароль (Password) и Подтверждение (Confirm password) введисти с клавиатуры пароль учетной записи пользователя.

4. Если необходима принудительная смена пароля при первой регистрации в сети, установить флажок Потребовать смену пароля при следующем входе в систему (User must change password at next logon). С целью защиты от атак по подбору пароля, следует установить срок действия пароля пользователя, сбросив флажок Срок действия пароля не ограничен (Password never expires).

6. Установленный флажок Запретить смену пароля пользователем (User cannot change password) запрещает пользователю самостоятельно изменять свой пароль.

7. Если только что созданная учетная запись по каким-либо причинам должна быть заблокирована, установить флажок Отключить учетную запись (Account disabled).

8. По завершении настройки создаваемой учетной записи нажмать кнопку Далее.

9. В окне диалога, запрашивающего подтверждение правильности выполняемого действия, нажмать кнопку Готово (Finish).

Для ввода дополнительной информации или изменения некоторых данных пользователя:

1. Указать учетную запись пользователя, информацию которой следует изменить, и нажмите правую кнопку мыши. В появившемся меню выберать команду Свойства.

2. Внести необходимые изменения и нажмите кнопку ОК.

Перемещение учетной записи пользователя.

Учетную запись пользователя можно перемещать из одного подразделения в другое в пределах одного домена или между доменами. Для соответствующего перемещения учетной записи этого пользователя следует воспользоваться технологией Drag and Drop (перетаскивание), применяемой практически ко всем визуальным объекта операционной системы семейства Windows.

Добавление пользователя в группу.

1. Указать группу, в которую необходимо добавить пользователя, и нажмите правую кнопку мыши. В появившемся меню выберать команду Свойства. Появится окно свойств группы.

2. Перейти на вкладку Члены группы (Members) окна свойств и нажать кнопку Добавить.

3. Появится окно Выбор: Пользователи, Контакты или Компьютеры (Select Users, Contacts, or Computers). Здесь можно задать область выполнения запроса: весь каталог, определенный домен или определенная часть дерева подразделения внутри домена. Обратите внимание, что каталог может состоять из множества доменов.

4. Выбрать имя добавляемого пользователя и нажать кнопку Добавить. Обратите внимание, что, нажав клавишу <Ctrl> и одновременно выполняя щелчки на нужных объектах, в этом диалоговом окне можно одновременно выбрать несколько пользователей или групп.

В результате все выбранные объекты станут членами соответствующей группы.

Удаленное управление компьютерами.

После создания объекта "компьютер" можно управлять им удаленно, диагностируя службы, работающие на этом компьютере, просматривая события и т. д.

Для того чтобы управлять компьютером удаленно:

1. В окне оснастки Active Directory- Пользователи и компьютеры укажите имя компьютера и нажать правую кнопку мыши. В появившемся меню выберать команду Управление (Manage).

2. Для выбранного компьютера будет запущена оснастка Управление компьютером (Computer Management).

Делегирование прав администрирования.

Как правило, сети больших предприятий на платформе Windows 2000 обладают чрезвычайно разветвленным деревом каталога. Большое количество ветвей, а также наличие достаточно автономных площадок организации, включенных в общее дерево каталога, усложняют управление. Администрирование сети,

каталог которой состоит из десятков тысяч объектов, не может безопасно осуществляться одним или несколькими администраторами, имеющими права доступа ко всем объектам.

В подобных случаях следует применять *делегирование прав администрирования*. Это чрезвычайно мощный инструмент, который в больших организациях позволяет более эффективно сконфигурировать систему безопасного администрирования. С его помощью управление отдельными областями сети смогут осуществлять специально назначенные ответственные лица - *администраторы*. При делегировании прав администрирования очень важно наделять ответственных лиц полномочиями, позволяющими выполнять функции администратора только в пределах их зоны ответственности, они не должны иметь возможность администрировать объекты каталога, находящиеся в других частях сети организации.

Права на создание новых пользователей или групп предоставляются на уровне подразделения или контейнера, в котором будут создаваться учетные записи. Администраторы групп одного подразделения могут не иметь прав на создание и управление учетными записями другого подразделения в том же домене. Однако, если права доступа и настройки политик получены на более высоком уровне дерева каталога, они могут распространяться вниз по дереву благодаря механизму *наследования прав доступа*.

Делегирование управления объектами групповой политики.

С помощью инструментов управления Active Directory администратор может делегировать другим пользователям и группам право управления частью каталога. Это в полной мере относится и к объектам групповой политики, в отношении которых могут быть, в частности, делегированы следующие права:

–управление связями GPO с сайтом, доменом или подразделением (организационной единицей). Для этого с помощью инструмента управления Active Directory следует указать объект (сайт, домен или организационную единицу) и щелкнуть правой кнопкой мыши. В появившемся контекстном меню выбрать команду Делегирование управления (Delegate Control). Запустится *Мастер делегирования управления* (Delegation of Control Wizard). С

его помощью можно выбрать объект групповой политики, группу или пользователя, которому должны быть делегированы права, а также и само право (в данном случае *Управление ссылками групповой политики* (Manage Group Policy links));

–создание и удаление всех дочерних объектов групповой политики. По умолчанию правом создания объектов в GPO обладают администраторы домена (Domain Admins) и администраторы предприятия (Enterprise Admins), а также операционная система. Для делегирования пользователю права управления объектами групповой политики домена необходимо включить его в группу «Создатели-владельцы групповой политики» (Group Policy Creator Owners);

–редактирование свойств объектов групповой политики. По умолчанию правом редактирования GPO обладают администраторы домена, администраторы предприятия и операционная система. Для делегирования пользователю права редактирования объекта групповой политики необходимо включить его в одну из указанных групп безопасности.

Чтобы позволить группе или пользователю управлять некоторым подразделением (контейнером):

1. Запустите Active Directory - Пользователи и компьютеры.

2. Укажите подразделение, управление которым необходимо передать, и нажмите правую кнопку мыши. В появившемся меню выберите команду Делегировать управление (Delegate control). Запустится Мастер делегирования управления (Delegation of Control Wizard). Нажмите кнопку Далее.

3. В следующем окне мастера нажмите кнопку Добавить и выберите пользователя или группу, которой вы хотите разрешить управление подразделением, нажмите кнопку ОК и затем кнопку Далее.

4. В открывшемся окне диалога мастера делегирования управления в окне со списком Делегировать следующие обычные задачи (Delegate the following common tasks) выберите одну или несколько операций, право выполнения которых делегируется указанному пользователю или группе. Если нужно делегировать право выполнения более специализированной задачи, установите

переключатель Создать особую задачу для делегирования (Create a custom task to delegate). Нажмите кнопку Далее.

5. Если указана особая задача для делегирования в следующем окне, можно выбрать область применения для этой задачи: положение переключателя Этой папкой и существующими в ней объектами, созданием новых объектов в этой папке (This folder, existing objects in this folder, and creation of new objects in this folder), в этом случае вы передадите группе право на администрирование всего контейнера, или положение Только следующими объектами в этой папке (Only the following objects in the folder) и установить флажки возле нужных объектов, в этом случае группа сможет управлять только *выбранными объектами.* Затем нажмите кнопку Далее.

6. В открывшемся окне определяются делегируемые разрешения. Можно отображать и устанавливать *общие* разрешения или разрешения для *отдельных* свойств или *дочерних* объектов. В пределах контейнера можно делегировать не все, а только некоторые права администрирования: например, можно делегировать только права на модификацию (чтение-запись) выбранного контейнера без дочерних объектов. Задайте нужные разрешения и нажмите кнопку Далее.

7. В следующем окне сводки выводится информация о выбранных действиях. Можно вернуться назад и скорректировать параметры. Если все правильно, нажмите кнопку Готово.

Эффективное функционирование многопользовательской операционной системы невозможно без четкого разграничения доступа к ресурсам. Одним из средств, позволяющих настраивать параметры безопасной работы пользователей в сети в операционных системах семейства Windows (NT, 2000, XP и выше), являются *политики безопасности.*

Реализация политик безопасности в Windows 2000 предоставляет достаточно широкие возможности, в том числе настройку политики безопасности для всего дерева доменов. Установив политику безопасности в одном месте, администраторы могут контролировать безопасность всех рабочих станций домена. Политики безопасности в Windows 2000 реализуются с помощью средств *групповых политик* (group policy).

Групповая политика имеет следующие преимущества:

–основываясь на службе Active Directory системы Windows 2000, позволяет как централизованно, так и децентрализованно управлять параметрами политики;

–обладает гибкостью и масштабируемостью. Может быть применена в широком наборе конфигураций системы, предназначенных как для малого бизнеса, так и для больших корпораций;

–обладает высокой степенью надежности и безопасности;

–групповые политики расширяют и используют преимущества Active Directory. Их настройки находятся в *объектах групповых политик* (Group Policy Object, GPO), которые в свою очередь ассоциируются с такими контейнерами Active Directory, как сайты, домены и подразделения (организационные единицы).

Для запуска объекта Групповая политика (рисунок 7.4) следует выполнить следующие действия:

1) выбрать объект Active Directory, для которого необходимо установить групповую политику безопасности (на рисунке 7.4 – для всего контролера домена) и правой кнопкой мыши вызвать контекстное меню;

2) выбрать элемент Свойства (Properties);

3) в диалоговом окне свойств выбрать вкладку Групповая политика (Group Policy);

4) для модификации глобальной политики следует выбрать кнопку Edit, если политика еще не была создана – New и ввести имя, либо воспользоваться тем, которое предлагает система.

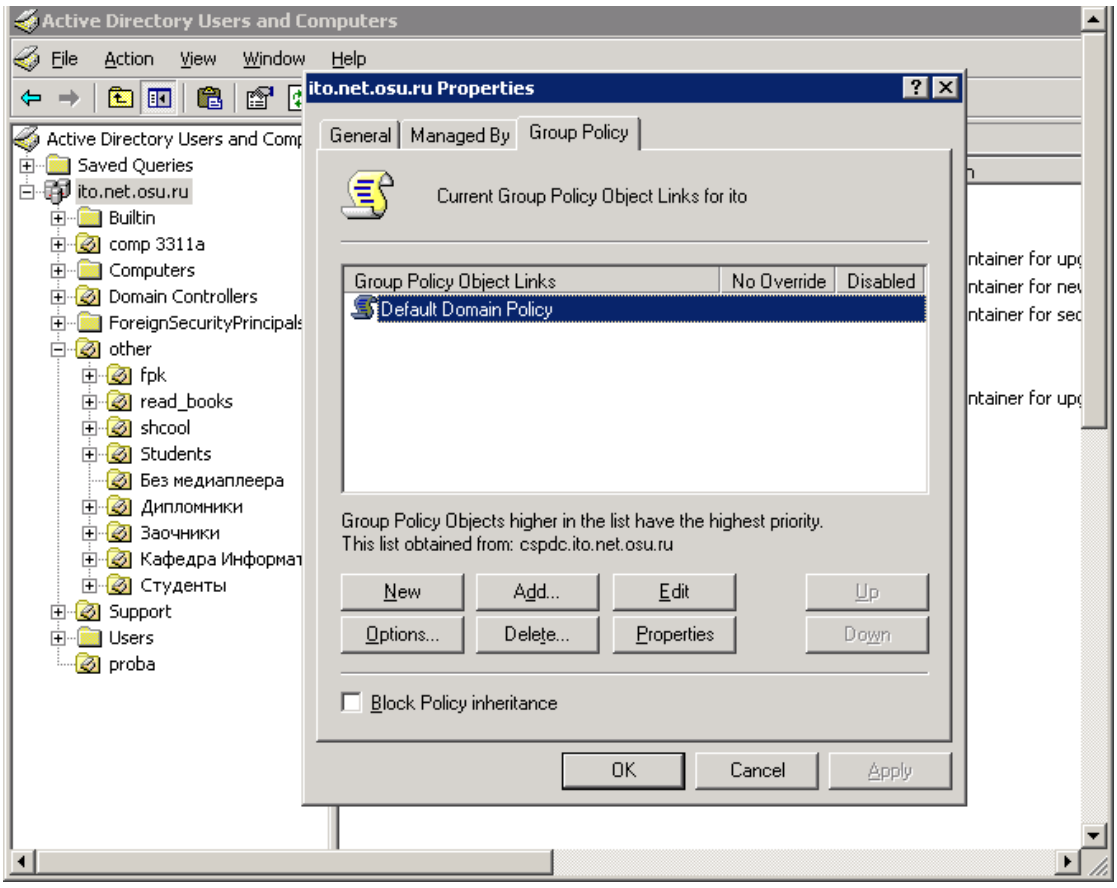

Рисунок 7.4

Создать групповую политику для контейнера Active Directory можно только при наличии определенного набора условий. Необходимо иметь работающий контроллер домена Windows 2000. Пользователь, который создает групповую политику, должен обладать правами на чтение и запись в системный том контроллеров домена (папка Sysvol). Кроме того, он должен иметь право модификации выбранного контейнера Active Directory.

После выбранных действий загружается корневой узел (рисунок 7.5), представляющий собой GPO, присоединенный к определенному контейнеру.

Имя этого GPO и имя контейнера, к которому он присоединен, отображаются в окне структуры в следующем формате:

Имя_политики [Имя_домена] Policy

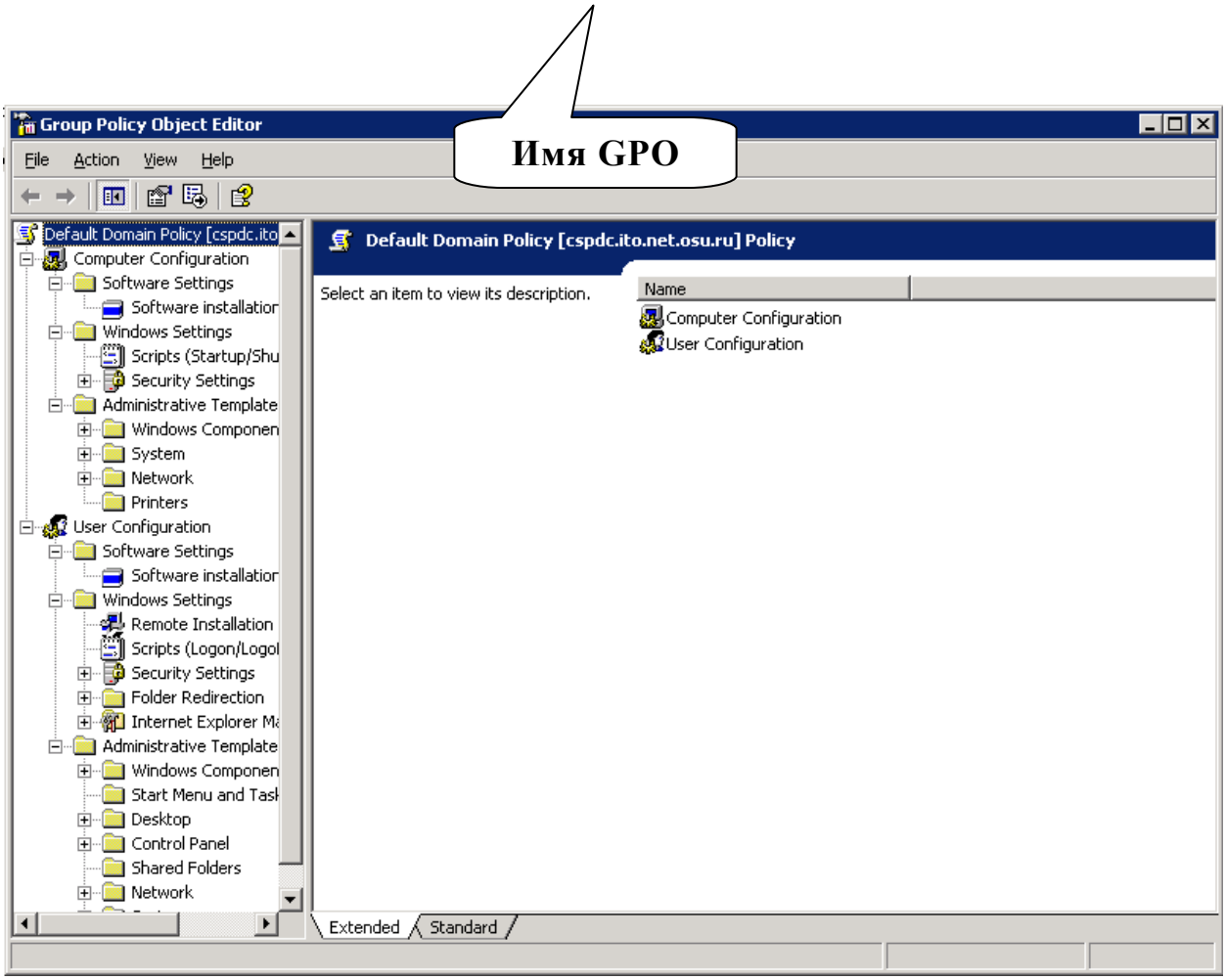

Рисунок 7.5

Затем пространство имен подразделяется на два узла более низкого уровня: «Конфигурация компьютера» (Computer Configuration) и «Конфигурация пользователя» (User Configuration). Используя их, можно создавать и настраивать групповые политики для компьютера и пользователей.

Узел «Конфигурация компьютера» содержит параметры всех политик, определяющих работу *компьютера*. Они регулируют функционирование операционной системы, вид рабочего стола, задают параметры выполняемых приложений, определяют работу средств обеспечения безопасности и т. д. Групповая политика применяется к рабочей станции домена на этапе загрузки системы и в дальнейшем при выполнении циклов обновления.

Узел «Конфигурация пользователя» содержит параметры всех политик, определяющих работу *пользователя* на компьютере. Они регулируют вид рабочего стола как и в предыдущем случае, задают параметры выполняющихся приложений, определяют

работу средств обеспечения безопасности и пользовательских сценариев входа и выхода. Групповая политика применяется к пользователю при его регистрации и в дальнейшем при выполнении циклов обновления.

Опишем некоторые расширения объекта «Групповая политика»:

–*административные шаблоны*. (Administrative Templates). Здесь находится групповая политика, определяющая параметры реестра, задающие работу и внешний вид рабочего стола, компонент операционной системы и приложений;

–*параметры безопасности* (Security Settings). Служит для настройки параметров системы безопасности компьютеров, на которые воздействует данный объект групповой политики. С помощью групповых политик можно настроить безопасность индивидуального компьютера, домена и целой сети;

–*установка программ* (Software Installation). Служит для централизованного управления программным обеспечением организации. С его помощью можно задавать различные режимы установки новых программ на компьютеры пользователей;

–*сценарии* (Scripts). Сценарии используются для автоматического выполнения набора команд при загрузке операционной системы и в процессе завершения ее работы, а также при регистрации и отключении пользователя от сети. Для выполнения сценариев, написанных на Microsoft JScript и Microsoft Visual Basic Scripting Edition, можно применять сервер сценариев (Windows Scripting Host);

–*перенаправление папок* (Folder Redirection). Позволяет перенаправлять обращение к специальным папкам в сеть.

С помощью расширения «Параметры безопасности» (рисунок 7.6) в GPO можно определить параметры политики безопасности, определяющие различные аспекты работы системы безопасности Windows 2000. Созданная в объекте групповой политики конфигурация воздействует на все компьютеры, находящиеся в контейнере, к которому присоединен данный GPO.

Расширение «Параметры безопасности» позволяет настраивать следующие аспекты системы безопасности компьютера:

–*политики учетных записей* (Account Policies). Можно настраивать политики безопасности как учетных записей в масштабах домена, так и локальных учетных записей. Здесь определяются политика паролей, политика блокировки паролей и политика Kerberos, распространяющаяся на весь домен;

–*локальные политики* (Local Policies). Можно настраивать политику аудита, назначать права пользователей и различные параметры безопасности, доступные для настройки в системе Windows 2000;

–*журнал событий* (Event Log). Можно настраивать политики безопасности, определяющие работу журналов событий приложений, системы и безопасности;

–*группы с ограниченным доступом* (Restricted Groups). Можно регулировать членство пользователей в специфических группах. Сюда обычно включают встроенные группы, такие как Администраторы, Операторы архива и другие, имеющие по умолчанию права администратора. В эту категорию могут быть включены и другие группы, безопасность которых требует особого внимания и членство в которых должно регулироваться на уровне политики;

–*системные службы* (System Services). Можно настраивать безопасность и параметры загрузки для работающих на компьютере служб. В этом разделе могут быть использованы расширения, с помощью которых можно осуществлять настройку безопасности, специфическую для данной службы. Например, расширение File Sharing Service позволяет настраивать политику безопасности для службы создания общего доступа к файлу (ограничение анонимного доступа к общим ресурсам, формирование безопасности различных сетевых общих ресурсов и т. д.);

–*реестр* (Registry). Можно настраивать безопасность различных разделов реестра;

–*файловая система* (File System). Можно настраивать безопасность определенных файлов;

–политики открытого ключа (Public Key Policies). Можно настраивать политики безопасности в отношении шифрования информации с помощью EFS, авторизации корневого сертификата в масштабах домена, авторизации доверенного сертификата и т. д.;

–политики безопасности IP (IPSEC). Позволяет настраивать политику безопасности IP для компьютеров.

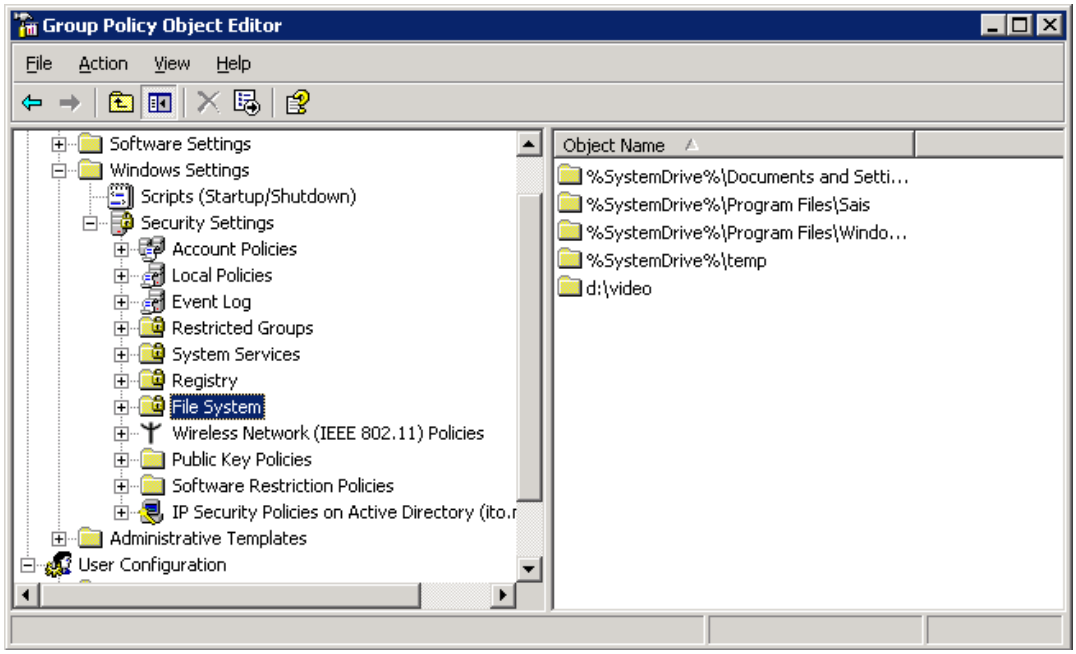

Рисунок 7.6

Политики безопасности, определяемые расширением «Параметры безопасности», действуют на компьютеры и частично на пользователей. Поскольку политика безопасности Windows 2000 значительно отличается от политик предыдущих версий Windows NT, при переходе к Windows 2000 низкоуровневые политики безопасности не переносятся. Если при переходе создается новое дерево доменов, одновременно создается и новая политика безопасности, назначаемая по умолчанию. Если при переходе домен присоединяется к уже существующему дереву, политика безопасности берется от родительского домена.

Для модификации настроек безопасности щелкните на папке «Параметры безопасности», затем щелчками на соответствующих узлах откройте весь путь, ведущий к интересующим настройкам. В правом подокне окна «Групповая политика» двойным щелчком

выберите настраиваемую политику и в открывшемся окне настройте ее.

Рассмотрим работу указанных расширений на конкретных примерах.

1. Настройка политики паролей.

Предположим, нам необходимо установить следующие правила политики паролей и блокировки:

–минимальная длина пароля – 8 символов;

–максимальный срок действия пароля – 30 дней;

–блокировать консоль после трех неудачных попыток входа.

Для реализации указанных правил выполним следующие действия:

1) откроем глобальную политику безопасности домена (см. выше) и расширение «Политики безопасности» (Security Settings);

2) выберем пункт «Политика учетных записей» (Account Policies), а затем политику паролей (Password Policy);

3) в правой части окна появится полный список правил, поддерживаемых политикой безопасности Windows 2000;

4) найдем требуемые правила:

а) минимальная длина пароля (Minimum password length);

б) максимальный срок действия пароля (Maximum password age);

установим требуемые значения, вызвав соответствующие диалоговые окна двойным щелчком мыши на названии правила;

5) для установки параметра блокировки перейдем в раздел политики блокировки учетных записей (Account Lockout Policy), выберем необходимое правило в правой части окна – Account lockout threshold – и установим требуемое значение – три.

2. Политика учетных записей.

Предположим, нам необходимо разрешить всем пользователям домена использовать привилегию изменения системного времени. Для этого следует выполнить:

1) откроем глобальную политику безопасности домена (см. выше) и расширение «Политики безопасности» (Security Settings);

2) выберем пункт «Локальные политики» (Local Policies), а затем политику назначения прав пользователей (User Rights Assignment);

3) в правой части окна выберем требуемое правило – Change the system time (рисунок 7.7) и добавим пользователя Все (Evryone).

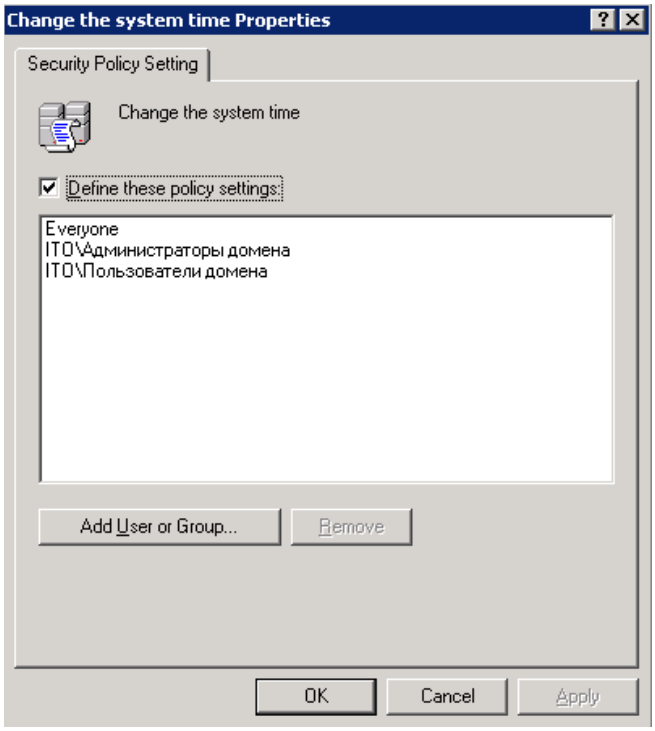

Рисунок 7.7

Применение групповых политик происходит в последовательности, соответствующей иерархии GPO: сначала объект групповой политики сайта, затем домена, затем GPO, связанные с подразделениями в соответствии с их вложенностью. Порядок выполнения групповых политик можно изменить с помощью настроек, блокирующих определенные групповые политики или заставляющих их выполняться принудительно. Кроме того, на порядок выполнения групповых политик влияет применение групп безопасности.

По умолчанию настройки групповой политики, применяемые к контейнеру определенного уровня, наследуются всеми

контейнерами более низких уровней и находящимися внутри них пользователями и компьютерами. Если с дочерней организационной единицей (контейнером) связан свой GPO, он может устанавливать для нее *индивидуальные* настройки групповых политик, отменяющие применение к ней *наследуемых* настроек. Если некоторые настройки групповых политик родительского контейнера не заданы (not defined), то они не наследуются и дочерними контейнерами. Если родительский контейнер обладает сконфигурированными настройками групповых политик, которые *не* заданы в GPO дочернего контейнера, то такие настройки наследуются.

Наследование настроек групповых политик родительского контейнера дочерним контейнером, с которым связан собственный объект групповой политики, может иметь место только в случае *совместимости* этих групповых политик. Например, если политика родительского контейнера задает определенную конфигурацию рабочего стола компьютера пользователя, а политика дочернего контейнера дополняет ее, пользователь увидит на своем рабочем столе все элементы, заданные обеими политиками. Если же групповая политика родительского контейнера противоречит групповой политике дочернего контейнера, выполняются только настройки GPO, связанного с дочерним контейнером.

Подобное положение вещей может быть изменено. Установка флажка Блокировать наследование политики (Block Policy inheritance), находящегося на вкладке Групповая политика окна свойств некоторого контейнера, *запрещает наследование* каких-либо групповых политик, установленных для *родительского* контейнера.

Существует средство, позволяющее установить *принудительное применение* групповой политики, настроенной для некоторого контейнера, всеми контейнерами *более низкого* уровня. Для этого на вкладке Групповая политика окна свойств контейнера следует нажать кнопку Параметры (Options). В появившемся окне диалога Параметры <*имя_подразделения*> необходимо установить флажок Не перекрывать (No override). В этом случае дочерние контейнеры будут наследовать (т. е. не смогут переопределить) все настройки родительского контейнера, даже в том случае, если для

дочерних контейнеров установлен флажок Блокировать наследование политики.

По умолчанию групповая политика применяется синхронно, т. е. политики компьютера применяются до появления окна «Вход в Windows» (Log on to Windows), а политики пользователя – до передачи операционной системой управления оболочке, интерактивно взаимодействующей с пользователем. Подобный порядок можно изменить, однако делать это не рекомендуется, поскольку асинхронное применение групповых политик может привести к непредсказуемым и нежелательным результатам.

Применение групповых политик не ограничивается только, например, моментом загрузки операционной системы компьютера или регистрацией пользователя в системе. При работе компьютера в сети групповые политики могут измениться, поэтому они применяются периодически (по умолчанию – каждые 90 минут). Длительность периода применения политик можно изменять. Если задать его равным нулю, групповые политики применяются через каждые 7 секунд. Следует учитывать, что при уменьшении периода применения групповых политик значительно увеличивается нагрузка на систему. На контроллерах доменов период применения политик равен 5 минутам.

Настройки расширений Установка программ и Переназначение папки применяются только при загрузке операционной системы или регистрации пользователя в системе, поскольку периодическое применение этих групповых политик может вызвать нежелательные результаты.

7.5.3 Вопросы и задания к лабораторной работе № 5

Вопросы:

1. Назначение Active Directory и основные возможности.

2. Какова структура Active Directory?

3. Для чего используются организационные единицы, когда и с какой целью их следует создавать?

4. Какие группы пользователей операционная система Windows 2000 создает по умолчанию?

5. Может ли один и тот же пользовать входить в разные группы?

6. В каких случаях следует использовать делегирование прав?

7. Какие права могут быть делегированы?

8. Пользователям каких групп можно делегировать права?

9. Какие виды политик безопасности поддерживаются в Windows 2000, сферы их применения?

10. Какие параметры безопасности можно настроить в глобальной политике безопасности?

11. Как взаимодействуют между собой глобальная и локальная политики безопасности?

12. Какие правила наследования политик безопасности поддерживаются?

Задания:

1. Создать новую организационную единицу (имя выбрать произвольно, например, my_unit).

2. Создать новую группу.

3. Создать в организационной единице трех новых пользователей: для всех потребовать смену пароля при входе и ограничить срок действия пароля. Одного из пользователей включить в новую группу.

4. Делегировать права на созданную организационную единицу пользователю из новой группы.

5. Установить максимальный срок действия пароля – 30 дней.

6. При вводе нового пароля требовать его неповторяемость. Хранить в системе 2 предыдущих пароля.

7. Установить минимальную длину пароля – 10 символов.

8. Установить аудит успеха для событий входа в систему.

9. Назначить возможность выключения системы только для администраторов.

10. Разрешить вход в систему только для членов группы "Администраторы" и пользователя "_____".

11. Разрешить доступ к компьютеру из сети только для пользователя "_____".

12. Блокировать консоль пользователя после ввода двух неверных паролей на 5 минут.

13. Отображать последнее имя пользователя при диалоге входа в систему.

14. Разрешить пользователю "_____" изменять политику аудита системы.

Список использованных источников

1. **Алферов А.П.** Основы криптографии / А.П. Алферов, А.Ю. Зубов, А.С. Кузьмин, А.В. Черемушкин. – М.:Гелиос АРВ, 2005. – 480 с.
2. **Баричев С.Г.** Основы современной криптографии. Учебный курс / С.Г. Баричев, В.В. Гончаров, Р.Е. Серов. – 2-е изд., испр. и доп. – М.: Горячая линия-Телеком, 2002. – 175 с.
3. Введение в криптографию / под общ. ред. В.В. Ященко. – М.: МЦНМО, «Черо», 1998.
4. **Виноградов И.М.** Элементы высшей математики: учеб. для вузов / И.М. Виноградов -М.: Высш. шк., 1999.
5. **Вистола А.** Отражение атак на уровне приложений / Альфредо Вистола // LAN – 2006. – № 2.
6. ГОСТ 28147-89 Системы обработки информации. Защита криптографическая. Алгоритм криптографического преобразования. -
7. ГОСТ 34.10-2001 Информационная технология. Криптографическая защита информации. Процессы формирования и проверки ЭЦП.
8. ГОСТ Р 34.11-94 Информационная технология. Криптографическая защита информации. Функция хэширования.
9. ГОСТ Р 50739-95 Средства вычислительной техники. Защиты от несанкционированного доступа к информации – М.: ИПК Издательство стандартов, 1995
10. ГОСТ Р 50922-96 Защита информации. Основные термины и определения – М.: ИПК Издательство стандартов, 1996
11. ГОСТ Р 52069.0-2003 Защита информации. Система стандартов. Основные положения – М.: ИПК Издательство стандартов, 2003
12. **Грушо А. А.** Теоретические основы защиты информации / А. А. Грушо, Е.Е. Тимонина – М.: "Яхтсмен", 1996. – 188с.
13. **Девянин П.Н.** Модели безопасности компьютерных систем: Учебное пособие для студ. высш. учеб. заведений / П.Н. Девянин– М.: Издательский центр «Академия», 2005. – 144 с.
14. **Девянин П.Н.** Теоретические основы компьютерной безопасности: Учебное пособие для вузов / П.Н. Девянин, О.О. Михальский, Д.И. Правиков – М.: Радио и связь, 2000. – 192 с.

15. **Домарев В.В. Защита информации и безопасность компьютерных систем** / В.В. Домарев. -Киев : Диа-Софт, 1999. - 480 с.
16. **Жельников В.** Криптография от папируса до компьютера / В. Жельников. -М. : ABF, 1996. - 336с.
17. **Зегжда Д.П.** Основы безопасности информационных систем: учеб. пособие / Д.П. Зегжда, А.М. Ивашко . - М. : Горячая линия - Телеком, 2000.
18. **Зубанов Ф.В.** Microsoft Windows 2000. Планирование, развертывание, установка / Ф.В. Зубанов. – 2-ое изд., испр. – М.: Издательско-торговый дом «Русская редакция», 2000. - 592 с.
19. **Конри-Мюррей Э.** Kerberos на страже безопасности компьютера / Э. Конри-Мюррей // LAN, 2001. – № 07-08.
20. **Коротыгин С.** IPSec — протокол защиты сетевого трафика на IP-уровне [Электронный ресурс] / С. Коротыгин. – Режим доступа: http://www.ixbt.com/comm/ipsecure.shtml
21. **Малюк А.А.** Введение в защиту информации в автоматизированных системах: Учебное пособие для вузов / А.А. Малюк, С.В. Пазизин, Н.С. Погожин – 2-е изд. – М.: Горячая линия – Телеком, 2004. – 147с.
22. **Малюк А.А.** Информационная безопасность: концептуальные и методологические основы защиты информации. Учеб. пособие для вузов. – М.: Горячая линия-Телеком, 2004. – 280 с.
23. **Мельников В.П.** Информационная безопасность и защита информации: учеб. пособие для студ. высш. учеб. заведений / В.П. Мельников, С.А. Клейменов, А.М. Петраков; под ред. С.А. Клейменова. – М.: Издательский центр «Академия», 2006. – 336 с.
24. **Немет Э.** Unix: руководство системного администратора / Э. Немет, Г. Снайдер, С. Сибасс, Т.Р. Хейн – 3-е изд. – СПб.: Питер, 2005. – 925с.
25. **Нечаев В.И.** Элементы криптографии. Основы теории защиты информации / В.И. Нечаев – М.: Высшая школа, 1999.

26. **Норткат С.** Анализ типовых нарушений безопасности в сетях / Купер М., Фирноу М. – Пер. с англ. – М.: Издательский дом «Вильямс», 2001. – 464с.
27. **Петренко С.А.** Политики информационной безопасности / С.А. Петренко, В.А. Курбатов. – М.: Компания АйТи, 2006. – 400 с.
28. Программирование алгоритмов защиты информации: Учебное пособие / ; А.В. Домашев, В.О. Попов, Д.И. Правиков и др. - М. : Нолидж, 2000 .
29. **Проскурин В.Г.** Программно-аппаратные средства обеспечения информационной безопасности. Защита в операционных системах: Учебное пособие для вузов / В.Г. Проскурин, С.В. Крутов, И.В. Мацкевич – М.: Радио и связь, 2000. – 168 с.
30. **Романец Ю.В.** Защита информации в компьютерных системах и сетях / Ю.В. Романец, П.А. Тимофеев, В.Ф. Шаньгин; под ред. В.Ф. Шаньгина.- 2-е изд., перераб. и доп. -М. : Радио и связь, 2001.
31. Руководящий документ. «АС. Защита от НСД к информации. Классификация АС и требования по защите информации», Гостехкомиссия России, 1998 г.
32. Руководящий документ. «Безопасность информационных технологий. Положение по разработке профилей защиты и заданий по безопасности», Гостехкомиссия России, 2003 г.
33. Руководящий документ. «Защита от несанкционированного доступа к информации. Термины и определения», Гостехкомиссия России, 1998 г.
34. Руководящий документ. «Концепция защиты СВТ и АС от НСД к информации», Гостехкомиссия России, 1998 г.
35. Руководящий документ. «СВТ. Защита от НСД к информации. Показатели защищенности от НСД к информации», Гостехкомиссия России, 1998 г.
36. Руководящий документ. «СВТ. МЭ. Защита от НСД к информации. Показатели защищенности от НСД к информации», Гостехкомиссия России, 1998 г.
37. **Соломон Д.** Внутреннее устройство Microsoft Windows 2000. Мастер-класс / Пер. с англ. / Д. Соломон, М. Руссинович — СПб.: Питер; М.: Издательско-торговый дом «Русская Редакция». 2004. —746 стр.
38. Спецификация протокола SSL версии 3 [Электронный ресурс] – Режим доступа: http://www.netscape.com/ newsref/std/SSL.html

39. **Столлингс В.** Криптография и защита сетей: принципы и практика / В. Столингс – 2-е изд.: Пер. с англ. – М.: Издательский дом «Вильямс», 2001. – 672с.
40. Теория и практика обеспечения информационной безопасности / под ред. П. Д. Зегжды. – М.: Издательство агентства "Яхтсмен", 1996. – 298с.
41. **Уэйнрайт П.** Apache для профессионалов / П. Уэйнрайт – М.: Издательство «Лори», 2001. – 474с.
42. **Шнайер Б.** Прикладная криптография / Б.Шнайер, М.: Триумф, 2002.
43. **Arnaud Aubert** TCP/IP и безопасность [Электронный ресурс] / Arnaud Aubert. – Режим доступа: http://www.securitylab.ru
44. **Artur Maj** Apache 2 с SSL/TLS: Шаг за Шагом, Часть 1 [Электронный ресурс] / Artur Maj. – Режим доступа: http://www.securitylab.ru
45. **Artur Maj** Apache 2 с SSL/TLS: Шаг за Шагом, Часть 2 [Электронный ресурс] / Artur Maj. – Режим доступа: http://www.securitylab.ru
46. **Dierks T.** The TLS Protocol Version 1.0 [Электронный ресурс] / Tim Dierks, Christopher Allen. – Режим доступа: http://ietf.org/rfc/rfc2246.txt
47. **Freier A.** The SSL Protocol Version 3.0 [Электронный ресурс] / Alan O. Freier, Philip Karlton, Paul C. Kocher. – Режим доступа: http://www.netscape.com/eng/ssl3
48. **Hickman K.** The SSL Protocol [Электронный ресурс] / Kipp E.B. Hickman. – Режим доступа: http://www.netscape.com/eng/security/SSL_2.html
49. **Kent S.** Security Architecture for the Internet Protocol [Электронный ресурс] / S. Kent, R. Atkinson. – Режим доступа: http://www.ietf.org/rfc/rfc2401.txt.