Методы строгой аутентификации:

- Стандарт X.509. Протоколы аутентификации с симметричными алгоритмами шифрования.
- Строгая аутентификация, основанная на асимметричных алгоритмах шифрования.
- Протокол аутентификации и распределения ключей Нидхэма-Шредера.

1. Аутентификация, авторизация и администрирование действий пользователей

Основные определения:

Идентификатор субъекта (пользователя) – связанная с субъектом некоторая информация, которая однозначно его идентифицирует.

Идентификация (Identification) – это процедура распознавания пользователя по его идентификатору (имени).

Аутентификация (Authentication) – это процедура проверки подлинности заявленного пользователя, процесса или устройства.

Авторизация (Authorization) – процедура предоставления субъекту определенных полномочий и ресурсов в данной системе (т.е. устанавливает сферу его действия и доступные ему ресурсы).

При защите каналов передачи данных должна выполняться взаимная аутентификация субъектов (т.е., взаимное подтверждение подлинности субъектов). Цель данной процедуры – обеспечить уверенность в том, что соединение установлено с законным субъектом и вся информация дойдет до места назначения. Обычно выполняется в начале сеанса связи в процессе установления соединения.

В зависимости от предъявляемых субъектом сущностей процессы аутентификации могут быть разделены на следующие категории:

1) на основе знания чего-либо (пароль, PIN–код, секретные и открытые ключи);

2) на основе обладания чем-либо (магнитные карты, смарт-карты, сертификаты и touch-memory);
3) на основе каких-либо неотъемлемых характеристик (биометрические характеристики пользователя – голос, сетчатки глаза, отпечатки пальцев, структура кровеносных сосудов ладони, рукописный почерк и т.д.)

Классификация процессов аутентификации по уровню обеспечиваемой безопасности:

1) аутентификация, использующая пароли и цифровые сертификаты;
2) строгая аутентификация на основе использования криптографических методов и средств;
3) процессы (протоколы) аутентификации, обладающие свойством доказательства с нулевым знанием;
4) биометрическая аутентификация пользователей.

Основные атаки на протоколы аутентификации:

1) маскарад (impersonation). Пользователь пытается выдать себя за другого с целью получения привилегий и возможности действий от лица этого пользователя;
2) подмена стороны аутентификационного обмена (interleaving attack). Злоумышленник в ходе данной атаки участвует в процессе аутентификационного обмена между двумя сторонами с целью модификации проходящего через него трафика;
3) повторная передача (replay attack). Заключается в повторной передаче аутентификационных данных каким-либо пользователем;

4) атака на основе подобранных сообщений (chosen-text attack). Злоумышленник перехватывает аутентификационный трафик и пытается получить информацию о долговременных криптографических ключах.

Основные способы предотвращения атак на протоколы аутентификации:

1) использование механизмов типа “запрос–ответ”, меток времени, случайных чисел, цифровых подписей;
2) привязка результата аутентификации к последующим действиям пользователя (например, создание секретных сеансовых ключей, которые используются при дальнейшем взаимодействии пользователей);
3) периодическое выполнение процедур аутентификации в рамках уже установленного сеанса связи.

Механизм “запрос–ответ” состоит в следующем. Пользователь *A* включает в посылаемое для пользователя *B* сообщение непредсказуемый элемент – запрос *X* (например, случайное число). При ответе пользователь *B* должен выполнить над этим элементом определённую операцию (например, вычислить некоторую функцию *f(x)*). Эту операцию невозможно выполнить заранее, так как пользователю B неизвестно, какое случайное число *X* придёт в запросе. Получив от пользователя *B* ответ с правильным результатом, пользователь *A* может быть уверен в подлинности пользователя *B*.

Механизм отметки времени подразумевает регистрацию времени для каждого сообщения. В этом случае каждый пользователь сети может определить, насколько устарело пришедшее сообщение и принять решение о его приёме.

Основные характеристики протоколов аутентификации:

1) наличие взаимной информации. Это свойство отражает необходимость обоюдной аутентификации между сторонами аутентификационного обмена;
2) вычислительная эффективность. Количество операций, необходимых для выполнения протокола;
3) коммуникационная эффективность. Данное свойство отражает количество сообщений и их длину, необходимые для осуществления аутентификации;
4) наличие третьей стороны. Примером третьей стороны может служить доверенный сервер распределения симметричных ключей или сервер, реализующий дерево сертификатов для распределения открытых ключей;
5) основа гарантий безопасности. Примером могут служить протоколы, обладающие свойством доказательств с нулевым знанием;
6) хранение секрета. Имеется в виду способ хранения критичной ключевой информации.

2. Методы аутентификации, использующие пароли и цифровые сертификаты

2.1. Аутентификация на основе многоразовых паролей (простая аутентификация).

В современных операционных системах (ОС) предусматривается централизованная служба аутентификации, которая выполняется одним из серверов сети и использует для своей работы базу данных. В этой базе данных хранятся учётные данные о пользователях сети, в которые включена информация о идентификаторах и паролях пользователей.

Процедура простой аутентификации пользователя в сети заключается в следующем. При попытке логического входа в сеть пользователь вводит свои идентификатор и пароль, которые поступают для обработки на сервер аутентификации. На сервере аутентификации производится сравнение введённой информации с хранящейся в базе данных и при её соответствии (совпадении) пользователь получает легальный статус.

Способы передачи пароля и идентификации пользователя:

1) в незашифрованном виде (например, PAP (Password Authentication Protocol) – протокол парольной аутентификации);
2) в защищённом виде. Все передаваемые данные (идентификатор и пароль пользователя, случайное число и метки времени) защищены посредством шифрования.

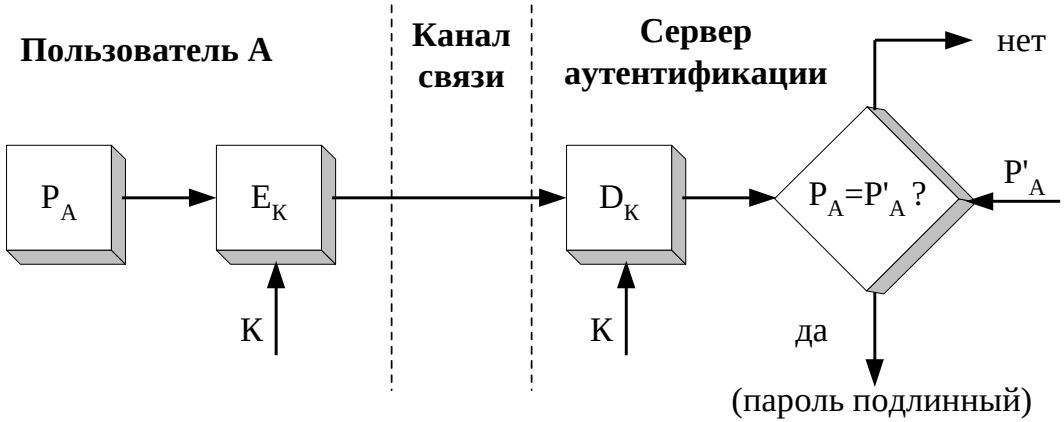


Рис. 1. Схема простой аутентификации с использованием пароля

где $E_к$ – средства шифрования и $D_к$ – расшифровывания;

P_A – пароль, введённый пользователем;

$P_A^{'}$ – исходное значение пароля, хранящегося на сервере аутентификации.

Схемы простой аутентификации отличаются также видом хранения и проверки паролей:

1) хранение паролей пользователей о открытом виде в системных файлах, защищённых от чтения и записи. Недостаток – возможность получения злоумышленником в системе привилегий администратора;
2) хранение и передача хэш-функций от паролей пользователей (использование односторонних функций). В этом случае гарантируется невозможность раскрытия пароля по его отображению, так как злоумышленник наталкивается на неразрешимую числовую задачу.

Вариант использования односторонней функции:

$$h(p)=E_p(ID)\ ,$$

где P – пароль пользователя;

ID – идентификатор пользователя;

E_P – процедура шифрования, выполняемая с использованием пароля P в качестве ключа.

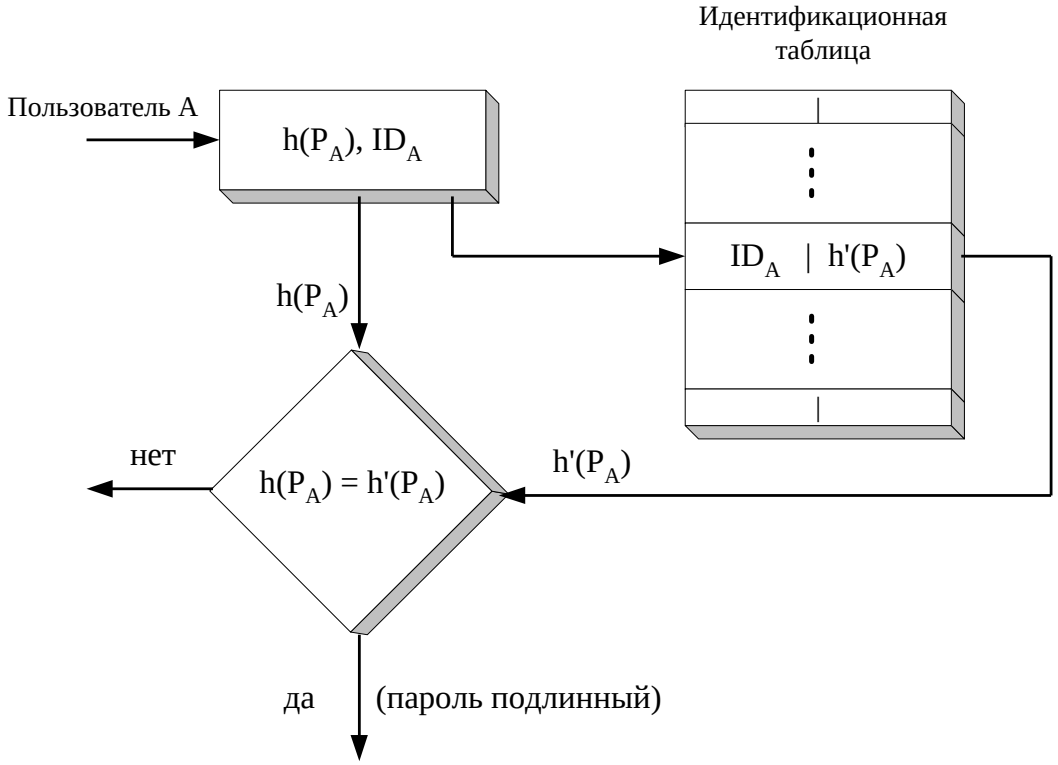


Рис. 2. Вариант использования односторонней функции

К протоколам аутентификации на основе многоразовых паролей относятся:

- PAP (Password Authentication Protocol);
- CHAP (Challenge–Handshake Authentication Protocol) – на основе процедуры “запрос–ответ”;
- TACACS (Terminal Access Controller Access Control System);
- RADIUS (Remote Authentication Dial–In User Service) – протоколы централизованного контроля доступа к сети удалённых пользователей.

Недостатки: схемы аутентификации, основанные на традиционных многоразовых паролях, не обладают достаточной безопасностью. Такие пароли можно перехватить, разгадать, подсмотреть или украсть.

2.2. Аутентификация на основе одноразовых паролей

Суть схемы одноразовых паролей – использование различных паролей при каждом новом запросе на предоставлении доступа. Одноразовый динамический пароль действителен только для одного входа в систему. Поэтому, даже если кто-то перехватил его, пароль окажется бесполезен.

Основные методы применения одноразовых паролей для аутентификации пользователей:

1. Использование механизма временн*ы*х меток на основе системы единого времени.
2. Использование общего для легального пользователя и проверяющего списка случайных

паролей и надёжного механизма их синхронизации.

3. Использование общего для пользователя и проверяющего генератора псевдослучайных чисел с одним и тем же начальным значением.

Один из наиболее известных протоколов аутентификации на основе одноразовых паролей – протокол S/Key (RFC.1760).

2.3. Аутентификация на основе сертификатов

При использовании цифровых сертификатов компьютерная сеть, которая даёт доступ к своим ресурсам, не хранит никакой информации о своих пользователях. Эту информацию пользователи предоставляют сами в своих запросах–сертификатах.

Сертификат представляет собой электронную форму, в которой содержится следующая информацию:

1) открытый ключ владельца данного сертификата;
2) сведения о владельце сертификата (например, имя, адрес электронной почты и т.д.);
3) наименование сертифицирующей организации, выдавшей данный сертификат;
4) электронная подпись сертифицирующей организации – зашифрованные закрытым ключом этой организации данные, содержащиеся в сертификате.

Цифровые сертификаты, удостоверяющие личность пользователя, выдаются по запросам

пользователей специальными уполномоченными организациями – центрами сертификации.

3. Строгая аутентификация

Идея строгой аутентификации заключается в следующем: Проверяемая (доказывающая) сторона доказывает свою подлинность проверяющей стороне, демонстрируя знание некоторого секрета. Доказательство знания секрета осуществляется с помощью последовательности запросов и ответов с использованием криптографических методов.

Существенным является тот факт, что доказывающая сторона демонстрирует только знание секрета, но сам секрет в ходе аутентификационного обмена не раскрывается.

В соответствии с рекомендациями стандарта X.509 различают процедуры строгой аутентификации следующих типов:

1) односторонняя аутентификация;
2) двусторонняя аутентификация;
3) трёхсторонняя аутентификация.

Односторонняя аутентификация предусматривает обмен информацией только в одном направлении.

Двусторонняя аутентификация по сравнению с односторонней содержит дополнительный ответ проверяющей стороны доказывающей стороне. Этот ответ должен убедить доказывающую сторону, что связь устанавливается именно с той стороной, которой были предназначены аутентификационные данные.

Трёхсторонняя аутентификация содержит дополнительную передачу данных от доказывающей стороны проверяющей стороне. Этот подход позволяет отказаться от использования меток времени при проведении аутентификации.

4. Строгая аутентификация, основанная на симметричных алгоритмах

4.1. Протоколы аутентификации с симметричными алгоритмами шифрования

Для работы протоколов аутентификации, построенных на основе симметричных алгоритмов шифрования, необходимо, чтобы проверяющий и доказывающий с самого начала имели один и тот же секретный ключ. Для закрытых систем с небольшим количеством пользователей каждая пара пользователей может заранее разделить его между собой. В больших распределённых системах часто используются протоколы аутентификации с участием доверенного сервера, с которым каждая сторона разделяет знание ключа.

Приведём примеры отдельных протоколов аутентификации, специфицированных в стандарте ISO/IEC 9798–2. При этом рассмотрим следующие варианты аутентификации:

1) односторонняя аутентификация с использованием меток времени;
2) односторонняя аутентификация с использованием случайных чисел;
3) двусторонняя аутентификация.

Введём следующие обозначения:

r_A – случайное число, сформированное участником *A*;

r_B – случайное число, сформированное участником *B*;

t_A – метка времени, сформированная участником *A*;

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

Односторонняя аутентификация, основанная на метках времени:

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

После получения и расшифровывания данного сообщения участник *B* убеждается в том, что метка времени t_A действительна и идентификатор *B*, указанный в сообщении, совпадает с его собственным. Предотвращение повторной передачи данного сообщения основывается на том, что без знания ключа невозможно оценить метку времени t_A и идентификатор *B*.

Односторонняя аутентификация, основанная на использовании случайных чисел:

$$A \leftarrow B: r_B$$

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

Участник *B* отправляет участнику *A* случайное число r_B. Участник *A* шифрует сообщение, состоящее из полученного числа r_B и идентификатора *B*, и отправляет зашифрованное сообщение участнику *B*. Участник *B* расшифровывает полученное сообщение и сравнивает полученную информацию с отправленной.

Двусторонняя аутентификация, использующая случайные значения:

$$A \leftarrow B: r_B \quad (1)$$

$$A \rightarrow B: E_K(r_A, r_B, B) \quad (2)$$

$$A \leftarrow B: E_K(r_A, r_B) \quad (3)$$

При получении второго сообщения участник *B* выполняет те же проверки, что и в предыдущем протоколе, и дополнительно расшифровывает случайное число r_A для включения его в третье сообщение для участника А. Третье сообщение, полученное участником *A*, позволяет ему убедиться в подлинности участника *B*.

4.2. Протоколы аутентификации, основанные на использовании однонаправленных ключевых хэш-функций

Протоколы, представленные выше, могут быть модифицированы путем замены симметричного шифрования на шифрование с помощью односторонней ключевой хэш-функции. Это бывает необходимо, если алгоритмы блочного шифрования недоступны или не отвечают предъявляемым требованиям (например, в случае экспортных ограничений).

Своеобразие шифрования с помощью односторонней хэш-функции заключается в том, что оно, по существу, является односторонним, то есть не сопровождается обратным преобразованием – расшифрованием на приемной стороне. Обе стороны (отправитель и получатель) используют одну и ту же

процедуру одностороннего шифрования.

Односторонняя хэш-функция h_K (.) с параметром-ключом K, примененная к шифруемым данным M, дает в результате хэш-значение m (дайджест), состоящее из фиксированного небольшого числа байтов (рис. 4.4).

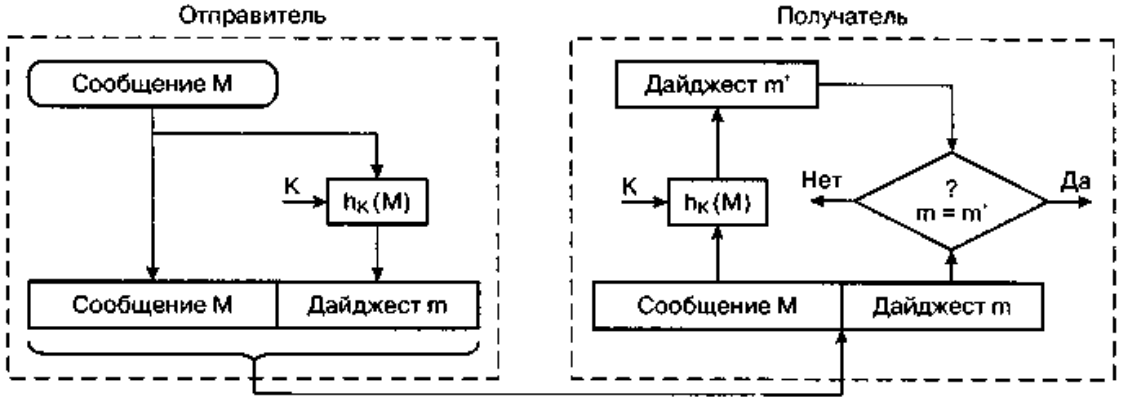


Рис. 4.3. Применение для аутентификации односторонней хэш-функции с параметром-ключом

Дайджест $m = h_k(M)$ передается получателю вместе с исходным сообщением M. Получатель сообщения, зная, какая односторонняя хэш-функция была применена для получения дайджеста, заново вычисляет ее, используя сообщение M. Если значения полученного дайджеста m и вычисленного дайджеста m' совпадают, значит, содержимое сообщения M не было подвергнуто никаким изменениям.

Знание дайджеста не дает возможности восстановить исходное сообщение, но позволяет проверить целостность данных. Дайджест представляет собой криптографически стойкую контрольную сумму для

исходного сообщения. Следовательно, между дайджестом и обычной контрольной суммой имеется существенное различие. Контрольную сумму используют как средство проверки целостности передаваемых сообщений по ненадежным линиям связи. Это средство проверки не рассчитано на борьбу со злоумышленниками, которым в такой ситуации ничто не мешает подменить сообщение, добавив к нему новое значение контрольной суммы. Получатель в таком случае не заметит никакой подмены.

В отличие от обычной контрольной суммы, при вычислении дайджеста применяются секретные ключи. В случае, если для получения дайджеста используется односторонняя хэш-функция с параметром-ключом *K*, который известен только отправителю и получателю, любая модификация исходного сообщения будет немедленно обнаружена.

При использовании для аутентификации односторонних функций шифрования в рассмотренные выше протоколы необходимо внести следующие изменения:

- функция симметричного шифрования $E_k(.)$ заменяется функцией $h_k(.)$;
- проверяющий, вместо установления факта совпадения значений в полях в расшифрованных сообщениях с предполагаемыми значениями, вычисляет значение однонаправленной функции и сравнивает его с полученным от другого участника обмена информацией;
- для обеспечения возможности независимого вычисления значения однонаправленной функции получателем сообщения в протоколе 1 метка времени t_A должна передаваться дополнительно в открытом виде, а в сообщении (2) протокола 3 случайное число r_A должно передаваться

дополнительно в открытом виде.

Модифицированный вариант протокола 3 с учетом сформулированных изменений имеет следующую структуру:

$$A \leftarrow B: r_B \quad (1)$$

$$A \rightarrow B: r_A,\ h_K(r_A, r_B, B) \quad (2)$$

$$A \leftarrow B: h_K(r_A, r_B, A) \quad (3)$$

Заметим, что в третье сообщение протокола включено поле A. Результирующий протокол обеспечивает взаимную аутентификацию и известен как протокол SKID 3.

4.3. Протокол аутентификации и распределения ключей Нидхэма–Шредера (Needham–Schroeder)

В данном случае участвуют две взаимодействующие стороны и доверенный сервер *KS*, выполняющий роль центра распределения ключей *KDC (Key Distribution Center)*.

Участники сеанса *A* и *B* имеют уникальные идентификаторы ID_A и ID_B соответственно. Стороны *A* и *B*, каждая по отдельности, разделяют свой секретный ключ с сервером *KS*.

Пусть сторона *A* хочет получить ключ сеансовый ключ для информационного обмена со стороной *B*.

Сторона A инициирует фазу распределения ключей, посылая по сети серверу *KS* идентификаторы ID_A и ID_B:

$$A \rightarrow KS: ID_A, ID_B \quad (1)$$

Сервер KS генерирует сообщение с временной отметкой *T*, сроком действия *L*, случайным сеансовым ключом *K* и идентификатором ID_A. Он шифрует это сообщение секретным ключом, который разделяет со стороной *B*.

Затем сервер KS берет *T*, *L*, *K*, идентификатор ID_B стороны *B* и шифрует полученное сообщение секретным ключом, который разделяет со стороной *A*.

Оба зашифрованные сообщения сервер *KS* отправляет стороне *A*:

$$KS \rightarrow A: E_A(T, L, K, ID_B),\ E_B(T, L, K, ID_A) \qquad (2)$$

Сторона A расшифровывает первое сообщение своим секретным ключом, проверяет отметку времени T , чтобы убедиться, что это сообщение не является повторением предыдущей процедуры распределения ключей. Затем сторона *A* генерирует сообщение со своим идентификатором ID_A и отметкой времени *T*, шифрует это сообщение сеансовым ключом *K* и отправляет стороне *B*. Кроме этого, *A* отправляет *B* вторую часть сообщения (2):

$$A \rightarrow B: E_K(ID_A, T),\ E_B(T, L, K, ID_A) \qquad (3)$$

Только сторона *B* может расшифровать сообщения (3). Сторона *B* получает отметку времени *T*, срок действия *L*, сеансовый ключ *K* и идентификатор ID_A. Затем сторона *B* расшифровывает сеансовым ключом *K* вторую часть сообщения (3). Совпадение значений *T* и ID_A в двух частях сообщения (3) подтверждают подлинность *A* по отношению к *B*.

Для взаимного подтверждения подлинности сторона *B* создает сообщение, состоящее из отметки

времени *T* плюс 1, шифрует его ключом *K* и отправляет стороне *A*:

$$B \rightarrow A : E_K(T+1) \quad (4)$$

Если после расшифровывания сообщения (4) сторона *A* получает ожидаемый результат, то она знает, что на другом конце линии связи действительно находится *B*.

Этот протокол успешно работает при условии, что часы каждого участника синхронизированы с часами сервера *KS*. Следует отметить, что в этом протоколе для получения сеансового ключа необходим обмен с сервером *KS* каждый раз, когда *A* желает установить связь с *B*. Протокол обеспечивает надежную аутентификацию при условии, что ни один из ключей не скомпрометирован и сервер *KS* защищен.

4.4. Протокол аутентификации и распределения ключей Kerberos.

Протокол Kerberos используется для аутентификации в системах «клиент-сервер» и обмена ключевой информации, предназначенной для установления защищенного канала связи между абонентами.

Kerberos обеспечивает аутентификацию в сетях, не заслуживающих доверия. То есть, при работе Kerberos подразумевается, что злоумышленники могут выполнять следующие действия:

- выдавать себя за одну из легитимных сторон;
- иметь физический доступ к одному из участвующих в соединении компьютеров;
- перехватывать любые пакеты, модифицировать их и передавать повторно.

Основу Kerberos составляет протокол аутентификации и распределения ключей *Нидхэма–Шредера* с третьей доверенной стороной.

Сервер Kerberos KS можно разделить на две части: сервер аутентификации *AS* (Authentication Server) и сервер службы выдачи мандатов *TGS* (Ticket Granting Service). Информационными ресурсами, доступ к которым хотят получить клиенты *C*, управляет сервер информационных ресурсов *RS*. Клиентами могут быть пользователи, а также независимые программы, выполняющие такие действия, как загрузка удаленных файлов, отправка сообщений, доступ к базам данных, доступ к принтерам и т.д. Предполагается, что серверы службы Kerberos надежно защищены от физического доступа злоумышленников.

Область действия системы Kerberos распространяется на тот участок сети, все пользователи которого зарегистрированы под своими именами и паролями в базе данных сервера Kerberos.

Принцип работы Kerberos:

В общих чертах процесс идентификации и аутентификации пользователя в системе Kerberos версии 5 можно описать следующим образом.

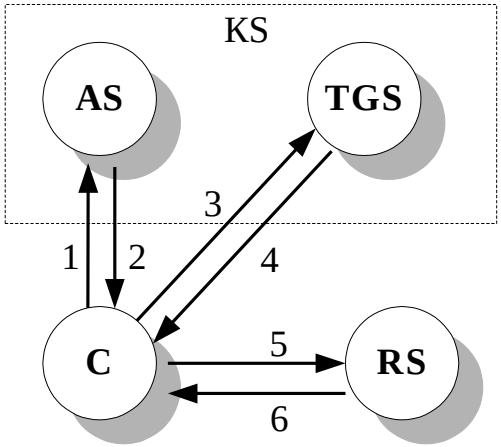
Клиент *C*, желая получит доступ к ресурсу сети, направляет запрос серверу аутентификации *AS*. Сервер AS идентифицирует пользователя с помощью его имени и пароля и высылает клиенту мандат на доступ к серверу службы выделения мандатов *TGS*.

Для использования конкретного целевого сервера информационных ресурсов *RS* клиент *C*

запрашивает у *TGS* мандат на обращение к целевому серверу *RS*. Если все в порядке, *TGS* разрешает использование необходимых ресурсов сети и посылает соответствующий мандат клиенту *C*.

Основные шаги работы системы Kerberos:

1. $C \rightarrow AS$ – запрос клиента *C* к серверу *AS* на разрешение обратиться к службе *TGS*.
2. $AS \rightarrow C$ – разрешение (мандат) от сервера *AS* клиенту *C* обратиться к службе *TGS*.
3. $C \rightarrow TGS$ – запрос клиента *C* к службе *TGS* на получение допуска (мандата) к серверу ресурсов *RS*.
4. $TGS \rightarrow C$ – разрешение (мандат) от службы *TGS* клиенту *C* для обращения к серверу ресурсов *RS*.
5. $C \rightarrow RS$ – запрос информационного ресурса (услуги) у сервера *RS*.
6. $RS \rightarrow C$ – подтверждение подлинности сервера *RS* и предоставление информационного ресурса (услуги) клиенту *C*.



Обозначения:
KS - сервер системы Kerberos
AS - сервер аутентификации
TGS - сервер службы выдачи мандатов
RS - сервер информационных ресурсов
C - клиент системы Kerberos

Рис. 4. Основные шаги работы системы Kerberos

Kerberos может использовать различные симметричные алгоритмы шифрования и хэш-функции, однако обязательными для поддержки установлены алгоритмы Triple DES и MD5.

5. Строгая аутентификация, основанная на асимметричных алгоритмах шифрования

В протоколах строгой аутентификации могут быть использованы асимметричные алгоритмы с открытыми ключами. В этом случае доказывающий может продемонстрировать знание секретного ключа одним из следующих способов:

- расшифровать запрос, зашифрованный на открытом ключе;
- поставить свою цифровую подпись на запросе.

Пара ключей, необходимая для аутентификации, не должна использоваться для других целей (например, для шифрования) по соображениям безопасности. Следует также предостеречь потенциальных пользователей о том, что выбранная система с открытым ключом должна быть устойчивой к атакам с выборкой шифрованного текста даже в том случае, если нарушитель пытается получить критичную информацию, выдав себя за проверяющего и действуя от его имени.

5.1. Аутентификация с использованием асимметричных алгоритмов шифрования

В качестве примера протокола, построенного на использовании асимметричного алгоритма шифрования, можно привести следующий протокол аутентификации:

$$A \leftarrow B: h(r),\ B,\ P_A(r, B) \quad (1)$$

$$A \rightarrow B: r \quad (2)$$

Участник *B* выбирает случайным образом число *r* и вычисляет значение $x = h(r)$ (значение *x* демонстрирует знание *r* без раскрытия самого значения *r*), далее он вычисляет значение $e = P_A(r, B)$. Под P_A подразумевается алгоритм асимметричного шифрования (например, RSA), а под *h(.)* – хэш-функция. Участник *B* отправляет сообщение (1) участнику *A*. Участник *A* расшифровывает $e = P_A(r, B)$ и получает значения r^1 и B^1, а также вычисляет $x^1 = h(r^1)$. После этого проводится ряд сравнений, доказывающих, что $x = x^1$ и что полученный идентификатор B^1 действительно указывает на участника *B*. В случае успешного проведения сравнения участник *A* посылает *r*. Получив его, участник *B* проверяет, то ли это значение, которое он отправил в первом сообщении.

В качестве следующего примера приведем модифицированный протокол Нидхэма и Шредера, основанный на асимметричном шифровании. Рассматривая вариант протокола Нидхэма и Шредера, используемый только для аутентификации, будем подразумевать под $e = P_B$ алгоритм шифрования открытым ключом участника *B*. Протокол имеет следующую структуру:

$$A \rightarrow B: P_B(r_1, A) \quad (1)$$

$$A \leftarrow B: P_A(r_2, r_1) \quad (2)$$

$$A \rightarrow B: r_2 \quad (3)$$

5.2. Аутентификация, основанная на использовании цифровой подписи

В рекомендациях стандарта X.509 специфицирована схема аутентификации, основанная на использовании цифровой подписи, меток времени и случайных чисел. Для описания данной схемы аутентификации введем следующие обозначения:

- t_A, r_A и r_B — временная метка и случайные числа соответственно;
- S_A — подпись, сгенерированная участником A;
- S_B — подпись, сгенерированная участником B;
- $cert_A$ — сертификат открытого ключа участника A;
- $cert_B$ — сертификат открытого ключа участника B.

Если участники имеют аутентичные открытые ключи, полученные друг от друга, тогда можно не пользоваться сертификатами, в противном случае они служат для подтверждения подлинности открытых ключей.

В качестве примеров приведем следующие протоколы аутентификации:

1. Односторонняя аутентификация с применением меток времени:

$$A \rightarrow B\colon\ cert_A,\ t_A,\ B,\ S_A(t_A, B) \qquad (1)$$

После принятия данного сообщения участник *B* проверяет правильность метки времени t_A, полученный идентификатор *B* и, используя открытый ключ из сертификата $cert_A$, корректность цифровой подписи $S_A(t_A,B)$.

2. Односторонняя аутентификация с использованием случайных чисел:

$$A \leftarrow B\colon\ r_B \tag{1}$$

$$A \rightarrow B\colon\ cert_A,\ r_A,\ B,\ S_A(r_A, r_B, B) \tag{2}$$

Участник *B*, получив сообщение от участника *A*, убеждается, что именно он является адресатом сообщения; используя открытый ключ участника *A*, взятый из сертификата $cert_A$, проверяет корректность подписи $S_A(r_A, r_B, B)$ под числом r_A, полученным в открытом виде, числом r_B, которое было отослано в первом сообщении, и его идентификатором *B*. Подписанное случайное число r_A используется для предотвращения атак с выборкой открытого текста.

3. Двусторонняя аутентификация с использованием случайных чисел:

$$A \leftarrow B\colon\ r_B \tag{1}$$

$$A \rightarrow B\colon\ cert_A,\ r_A,\ B,\ S_A(r_A, r_B, B) \tag{2}$$

$$A \leftarrow B\colon\ cert_B,\ A,\ S_B(r_A, r_B, A) \tag{3}$$

В данном протоколе обработка сообщений (1) и (2) выполняется так же, как и в предыдущем протоколе, а сообщение (3) обрабатывается аналогично сообщению (2).