Лекция №18. Технологии аутентификации.

Оглавление

[Введение 1](#_Toc508952359)

[1.Идентификация и аутентификация пользователей и программ 1](#_Toc508952360)

[2.Методы аутентификации, использующие пароли и PIN-коды 6](#_Toc508952361)

[2.1.Аутентификация на основе многоразовых паролей 7](#_Toc508952362)

[2.2.Аутентификация на основе одноразовых паролей 10](#_Toc508952363)

[3.Строгая аутентификация 15](#_Toc508952364)

[3.1.Основные понятия 15](#_Toc508952365)

[3.2.Строгая аутентификация, основанная на симметричных алгоритмах 16](#_Toc508952366)

[3.3.Строгая аутентификация, основанная на асимметричных алгоритмах 19](#_Toc508952367)

# Введение

Применение открытых каналов передачи данных создает потенциальные возможности для действий злоумышленников (нарушителей).

Поэтому одной из важных задач обеспечения информационной безопасности при взаимодействии пользователей является использование методов и средств, позволяющих одной (проверяющей) стороне убедиться в подлинности другой (проверяемой) стороны. Обычно для решения данной проблемы применяются специальные приемы, дающие возможность проверить подлинность проверяемой стороны.

# 1.Идентификация и аутентификация пользователей и программ

С каждым зарегистрированным в компьютерной системе субъектом (пользователем или процессом, действующим от имени пользователя) связана некоторая информация, однозначно идентифицирующая его. Это может быть число или строка символов, именующие данный субъект. Эту информацию называют идентификатором субъекта.

Операционная система Windows для идентификации использует специальный идентификатор безопасности (Security Identifier, SID), который присваивается каждой учетной записи пользователя (а также компьютера или группы) при ее создании. В отличие от имени каждый SID уникален, что позволяет системе однозначно идентифицировать пользователя. Поэтому операционная система оперирует именно SID-ами и использует их для контроля доступа к различным ресурсам — принтерам, файлам и папкам и т.п.

Примечание. Говоря об уникальности SID, надо сделать одну оговорку. В Windows существуют так называемые встроенные (BuiltIn) учетные записи, такие как Administrator или Guest. SID-ы этих записей одинаковы в каждом экземпляре Windows вне зависимости от версии ОС. Это дает администратору возможность более-менее централизованно управлять доступом при отсутствии доменной структуры.

SID для локальных учетных записей храниться базе данных диспетчера учетных записей (Security Account Manager, SAM) на локальном компьютере, для доменных — в базе Active Directory. И сегодня речь пойдет о том, как можно узнать SID пользователя по его имени и наоборот, как выяснить имя пользователя по его SID.

Когда требуется посмотреть SID текущего пользователя, то проще всего воспользоваться утилитой whoami. Для этого надо открыть консоль cmd и выполнить команду: **whoami /user.**



В Unix-подобных операционных системах пользователи идентифицируются идентификаторами пользователя (англ. User identifier, UID).

Операционная система различает пользователей именно по UID (а не, например, по логину). Во многих системах существует возможность создать две записи пользователя с разными логинами, но одинаковыми UID; в результате оба логина будут иметь одинаковые права, так как с точки зрения системы они неотличимы (так как имеют одинаковый UID). Это может использоваться злоумышленниками: проникнув в систему и получив права root, взломщик может создать себе аккаунт с UID=0, чтобы потом возвращаться в систему под логином, не привлекающим внимания, но получать права root.

Множество допустимых значений UID зависит от системы; в общем случае UID допускает использование значений от 0 до 65535 с некоторыми оговорками:

Суперпользователь всегда должен иметь UID, равный нулю (0).

Пользователю nobody обычно присваивается или наибольший из возможных UID (в противоположность суперпользователю), или один из системных UID (см. ниже).

UIDы с 1 по 100 по соглашению резервируются под системные нужды; некоторые руководства рекомендуют резервировать UIDы со 101 по 499 (в Red Hat) или даже 999 (в Debian).

Значение UID ставится в соответствие пользователю в файле /etc/passwd.

Если пользователь имеет идентификатор, зарегистрированный в сети, он считается легальным (законным) пользователем; остальные пользователи относятся к нелегальным пользователям. Прежде чем получить доступ к ресурсам компьютерной системы, пользователь должен пройти процесс первичного взаимодействия с компьютерной системой, который включает идентификацию и аутентификацию.

**Идентификация (Identification) — процедура распознавания пользователя по его идентификатору (имени).** Эта функция выполняется, когда пользователь делает попытку войти в сеть. Пользователь сообщает системе по ее запросу свой идентификатор, и система проверяет в своей базе данных его наличие.

Процедура идентификации напрямую связана с аутентификацией: субъект проходит процедуру аутентификации, и если аутентификация успешна, то информационная система на основе факторов аутентификации определяет идентификатор субъекта. При этом достоверность идентификации полностью определяется уровнем достоверности выполненной процедуры аутентификации.

**Аутентификация (Authentication) — процедура проверки подлинности заявленного пользователя, процесса или устройства.** Эта проверка позволяет достоверно убедиться, что пользователь (процесс или устройство) является именно тем, кем себя объявляет. При проведении аутентификации проверяющая сторона убеждается в подлинности проверяемой стороны, при этом проверяемая сторона тоже активно участвует в процессе обмена информацией. Обычно пользователь подтверждает свою идентификацию, вводя в систему уникальную, не известную другим пользователям информацию о себе (например, пароль или сертификат).

Идентификация и аутентификация являются взаимосвязанными процессами распознавания и проверки подлинности субъектов (пользователей). Именно от них зависит последующее решение системы: можно ли разрешить доступ к ресурсам системы конкретному пользователю или процессу. После идентификации и аутентификации субъекта выполняется его авторизация.

**Авторизация (Authorization) — процедура предоставления субъекту определенных полномочий и ресурсов в данной системе.** Иными словами, авторизация устанавливает сферу его действия и доступные ему ресурсы. Если система не может надежно отличить авторизованное лицо от неавторизованного, то конфиденциальность и целостность информации в этой системе могут быть нарушены. Организации необходимо четко определить свои требования к безопасности, чтобы принимать решения о соответствующих границах авторизации.

Авторизацию не следует путать с аутентификацией: аутентификация — это процедура проверки легальности пользователя или данных, например, проверки соответствия введённого пользователем пароля к учётной записи паролю в базе данных, или проверка цифровой подписи письма по ключу шифрования, или проверка контрольной суммы файла на соответствие заявленной автором этого файла. Авторизация же производит контроль доступа легальных пользователей к ресурсам системы после успешного прохождения ими аутентификации. Зачастую процедуры аутентификации и авторизации совмещаются.

С процедурами аутентификации и авторизации тесно связана процедура администрирования действий пользователя.

**Администрирование (Accounting) — регистрация действий пользователя в сети, включая его попытки доступа к ресурсам.** Хотя эта учетная информация может быть использована для выписывания счета, с позиций безопасности она особенно важна для обнаружения, анализа инцидентов безопасности в сети и соответствующего реагирования на них. Записи в системном журнале, аудиторские проверки и ПО accounting — все это может быть использовано для обеспечения подотчетности пользователей, если что-либо случится при входе в сеть с их идентификатором.

Необходимый уровень аутентификации определяется требованиями безопасности, которые установлены в организации. Общедоступные Web-серверы могут разрешить анонимный или гостевой доступ к информации. Финансовые транзакции могут потребовать строгой аутентификации. Примером слабой формы аутентификации может служить использование IP-адреса для определения пользователя. Подмена (spoofing) IP-адреса может легко разрушить механизм аутентификации. Надежная аутентификация является тем ключевым фактором, который гарантирует, что только авторизованные пользователи получат доступ к контролируемой информации.

При защите каналов передачи данных должна выполняться взаимная аутентификация субъектов, т. е. взаимное подтверждение подлинности субъектов, связывающихся между собой по линиям связи. Процедура подтверждения подлинности выполняется обычно в начале сеанса установления соединения абонентов. Термин «соединение» указывает на логическую связь (потенциально двустороннюю) между двумя субъектами сети. Цель данной процедуры — обеспечить уверенность, что соединение установлено с законным субъектом, и вся информация дойдет до места назначения.

Для подтверждения своей подлинности субъект может предъявлять системе разные сущности. В зависимости от предъявляемых субъектом сущностей процессы аутентификации могут быть разделены на основе:

* *знания* чего-либо. Примерами могут служить пароль, персональный идентификационный код PIN (Personal Identification Number), а также секретные и открытые ключи, знание которых демонстрируется в протоколах типа запрос–ответ;
* *обладания* чем-либо. Обычно это магнитные карты, смарт- карты, сертификаты и устройства *touch memory,*
* *каких-либо неотъемлемых* *характеристик.* Эта категория включает методы, базирующиеся на проверке *биометрических характеристик пользователя* (голоса, радужной оболочки и сетчатки глаза, отпечатков пальцев, геометрии ладони и др.). В данной категории не используются криптографические методы и средства. Аутентификация на основе биометрических характеристик применяется для контроля доступа в помещения или к какой-либо технике.

Пароль — это то, что знает пользователь и другой участник взаимодействия. Для взаимной аутентификации участников взаимодействия может быть организован обмен паролями между ними.

Персональный идентификационный номер PIN (Personal Identification Number) является испытанным способом аутентификации держателя пластиковой карты и смарт-карты. Секретное значение PIN-кода должно быть известно только держателю карты.

Динамический (одноразовый) пароль — это пароль, который после однократного применения никогда больше не используется. На практике обычно используется регулярно меняющееся значение, которое базируется на постоянном пароле или ключевой фразе.

Система запрос–ответ. Одна из сторон инициирует аутентификацию с помощью посылки другой стороне уникального и непредсказуемого значения «запрос», а другая сторона посылает ответ, вычисленный с помощью «запроса» и секрета. Так как обе стороны владеют одним секретом, то первая сторона может проверить правильность ответа второй стороны.

Сертификаты и цифровые подписи. Если для аутентификации используются сертификаты, то требуется применение цифровых подписей на этих сертификатах. Сертификаты выдаются ответственным лицом в организации пользователя, сервером сертификатов или внешней доверенной организацией. В рамках Интернета появились коммерческие инфраструктуры управления открытыми ключами PKI (Public Key infrastructure) для распространения сертификатов открытых ключей. Пользователи могут получить сертификаты различных уровней.

Процессы аутентификации можно также классифицировать по уровню обеспечиваемой безопасности. В соответствии с этим процессы аутентификации разделяются на следующие типы:

* аутентификация, использующая пароли и PIN-коды;
* строгая аутентификация на основе использования криптографических методов и средств;
* биометрическая аутентификация пользователей.

С точки зрения безопасности каждый из перечисленных типов способствует решению своих специфических задач, поэтому процессы и протоколы аутентификации активно используются на практике.

Основные атаки на протоколы аутентификации:

* + маскарад (Impersonation). Пользователь выдает себя за другого с целью получения полномочий и возможности действий от лица другого пользователя;
	+ 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;
	+ повторная передача (Replay attack) заключается в повторной передаче аутентификационных данных каким-либо пользователем;
	+ принудительная задержка (Forced delay). Злоумышленник перехватывает некоторую информацию и передает ее спустя некоторое время;
	+ атака с выборкой текста (Chosen-text attack). Злоумышленник перехватывает аутентификационный трафик и пытается получить информацию о долговременных криптографических ключах.

Для предотвращения таких атак при построении протоколов аутентификации применяются:

* + использование механизмов типа «запрос—ответ», «отметка времени», случайных чисел, идентификаторов, цифровых подписей;
	+ привязка результата аутентификации к последующим действиям пользователей в рамках системы. Примером подобного подхода может служить осуществление в процессе аутентификации обмена секретными сеансовыми ключами, которые используются при дальнейшем взаимодействии пользователей;
	+ периодическое выполнение процедур аутентификации в рамках уже установленного сеанса связи и т. п.

Механизм «запрос–ответ» состоит в следующем. Если пользователь А хочет быть уверенным, что сообщения, получаемые им от пользователя В, не являются ложными, он включает в посылаемое для В сообщение непредсказуемый элемент — запрос X (например, некоторое случайное число). При ответе пользователь В должен выполнить над этим элементом некоторую операцию (например, вычислить некоторую функцию f(X)). Это невозможно осуществить заранее, так как пользователю В неизвестно, какое случайное число X придет в запросе. Получив ответ с результатом действий В, пользователь А может быть уверен, что В — подлинный. Недостаток этого метода — возможность установления закономерности между запросом и ответом.

Механизм «отметка времени» подразумевает регистрацию времени для каждого сообщения. В этом случае каждый пользователь сети определяет, насколько «устарело» пришедшее сообщение, и решает 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.

При использовании отметок времени возникает проблема допустимого временного интервала задержки для подтверждения подлинности сеанса: сообщение с «временным штемпелем» в принципе не может быть передано мгновенно. Кроме того, компьютерные часы получателя и отправителя не могут быть абсолютно синхронизированы.

При сравнении и выборе протоколов аутентификации необходимо учитывать следующие характеристики:

* + наличие взаимной аутентификации. Это свойство отражает необходимость обоюдной аутентификации между сторонами аутентификационного обмена;
	+ вычислительную эффективность. Это количество операций, необходимых для выполнения протокола;
	+ коммуникационную эффективность. Данное свойство отражает количество сообщений и их длину, необходимую для осуществления аутентификации;
	+ наличие третьей стороны. Примером третьей стороны может служить доверенный сервер распределения симметричных ключей или сервер, реализующий дерево сертификатов для распределения открытых ключей;
	+ гарантии безопасности. Примером может служить применение шифрования и цифровой подписи.

# 2.Методы аутентификации, использующие пароли и PIN-коды

Одной из распространенных схем аутентификации является простая аутентификация, которая основана на применении традиционных многоразовых паролей с одновременным согласованием средств его использования и обработки. Аутентификация на основе многоразовых паролей — простой и наглядный пример использования разделяемой информации. Пока в большинстве защищенных виртуальных сетей VPN (Virtual Private Network) доступ клиента к серверу разрешается по паролю.

Однако все чаще применяются более эффективные средства аутентификации, например, программные и аппаратные системы аутентификации на основе одноразовых паролей, смарт-карт, USB-токенов, PIN-кодов, цифровых сертификатов, программные и аппаратные системы аутентификации на основе одноразовых паролей.

## 2.1.Аутентификация на основе многоразовых паролей

Базовый принцип «единого входа» предполагает достаточность одноразового прохождения пользователем процедуры аутентификации для доступа ко всем сетевым ресурсам. Поэтому в современных операционных системах предусматривается централизованная служба аутентификации, которая выполняется одним из серверов сети и использует для своей работы базу данных (БД). В этой БД хранятся учетные данные о пользователях сети, включающие идентификаторы (login) и пароли (password) пользователей, а также другую информацию.

Примером такой БД является Active Directory. Active Directory («Активный каталог», AD) — службы каталогов корпорации Microsoft для операционных систем семейства Windows Server. Первоначально создавалась, как LDAP-совместимая реализация службы каталогов, однако, начиная с Windows Server 2008, включает возможности интеграции с другими службами авторизации, выполняя для них интегрирующую и объединяющую роль.

Процедуру простой аутентификации пользователя в сети можно представить следующим образом. Пользователь при попытке логического входа в сеть набирает свои идентификатор и пароль. Эти данные поступают для обработки на сервер аутентификации. В БД, хранящейся на сервере аутентификации, по идентификатору пользователя находится соответствующая запись. Из нее извлекается пароль и сравнивается с тем паролем, который ввел пользователь. Если они совпали, то аутентификация прошла успешно — пользователь получает легальный статус и получает те права и ресурсы сети, которые определены для его статуса системой авторизации.

В схеме простой аутентификации (рис. 1) передача пароля и идентификатора пользователя может производиться следующими способами:

* в незашифрованном виде: например, согласно протоколу парольной аутентификации PAP (Password Authentication Protocol) пароли передаются по линии связи в открытой незащищенной форме;
* в защищенном виде: все передаваемые данные (идентификатор и пароль пользователя, случайное число и метки времени) защищены посредством шифрования или однонаправленной функции.



*Рис. 1. Простая аутентификация с использованием пароля.*

Очевидно, что вариант аутентификации с передачей пароля пользователя в незашифрованном виде не гарантирует даже минимального уровня безопасности, так как подвержен многочисленным атакам и легко компрометируется. Чтобы защитить пароль, его нужно зашифровать перед пересылкой по незащищенному каналу. Для этого в схему включены средства шифрования *Ек* и расшифровывания *DK,* управляемые разделяемым секретным ключом *К.* Проверка подлинности пользователя основана на сравнении присланного пользователем пароля *РА* и исходного значения *Р’А*, хранящегося на сервере аутентификации. Если значения *РА* и *Р'А* совпадают, то пароль *РА* считается подлинным, а пользователь *А* — законным.

Схемы организации простой аутентификации отличаются не только методами передачи паролей, но и видами их хранения и проверки. Наиболее распространенным способом является хранение паролей пользователей в открытом виде в системных файлах, причем на эти файлы устанавливаются атрибуты защиты от чтения и записи (например, при помощи описания соответствующих привилегий в списках контроля доступа ОС). Система сопоставляет введенный пользователем пароль с хранящейся в файле паролей записью. При этом способе не используются криптографические механизмы, такие как шифрование или однонаправленные функции. Очевидным недостатком этого способа является возможность получения злоумышленником в системе привилегий администратора, включая права доступа к системным файлам, и в частности, к файлу паролей.

Для обеспечения надежной защиты ОС пароль каждого пользователя должен быть известен только этому пользователю и никому другому, в том числе и администраторам системы. На первый взгляд то, что администратор знает пароль некоторого пользователя, не отражается негативно на безопасности системы, поскольку администратор, войдя в систему от имени обычного пользователя, получает права меньшие чем те, которые он получит, зайдя в систему от своего имени. Однако, входя в систему от имени другого пользователя, администратор получает возможность обходить систему аудита, а также совершать действия, компрометирующие этого пользователя, что недопустимо в защищенной системе. Таким образом, пароли пользователей не должны храниться в ОС в открытом виде.

С точки зрения безопасности предпочтительным является метод передачи и хранения паролей с использованием односторонних функций. Обычно для шифрования паролей в списке пользователей используют одну из известных криптографически стойких *хэш-функций.* В списке пользователей хранится не сам пароль, а *образ пароля,* являющийся результатом применения к паролю хэш-функции.

**Хэширование или хеширование (англ. hashing) — преобразование массива входных данных произвольной длины в (выходную) битовую строку фиксированной длины, выполняемое определённым алгоритмом. Функция, реализующая алгоритм и выполняющая преобразование, называется «хеш-функцией» или «функцией свёртки». Исходные данные называются входным массивом, «ключом» или «сообщением». Результат преобразования (выходные данные) называется «хешем», «хеш-кодом», «хеш-суммой», «сводкой сообщения».**

Однонаправленность хэш-функции не позволяет восстановить пароль по образу пароля, но позволяет, вычислив хэш-функцию, получить образ введенного пользователем пароля и таким образом проверить правильность введенного пароля. В простейшем случае в качестве хэш-функции используется результат шифрования некоторой константы на пароле.

Например, односторонняя функция h(**.**) может быть определена следующим образом:

$h\left(P\right)=E\_{P}\left(ID\right)$,

где *Р* — пароль пользователя; *ID* — идентификатор пользователя; *ЕР* — процедура шифрования, выполняемая с использованием пароля *Р* в качестве ключа.

Такие функции удобны, если длина пароля и ключа одинаковы. В этом случае проверка подлинности пользователя *А с* помощью пароля *РA* состоит из пересылки серверу аутентификации отображения *h(PA)* и сравнения его с предварительно вычисленным и хранимым в БД сервера аутентификации эквивалентом *h'(PA)* (рис. 2). Если отображения *h(PA)* и *h'(PA)* равны, то считается, что пользователь успешно прошел аутентификацию.



*Рис. 2. Использование односторонней функции для проверки пароля*

Системы простой аутентификации на основе многоразовых паролей имеют пониженную стойкость, поскольку выбор аутентифицирующей информации происходит из относительно небольшого числа слов. Срок действия многоразового пароля должен быть определен в политике безопасности организации. Пароли должны регулярно изменяться, быть трудными для угадывания и не присутствовать в словаре.

Схемы аутентификации, основанные на многоразовых паролях, не обладают достаточной безопасностью. Такие пароли можно перехватить, разгадать, подсмотреть или просто украсть. Более надежными являются процедуры аутентификации на основе одноразовых паролей.

## 2.2.Аутентификация на основе одноразовых паролей

Как уже отмечалось, схемы аутентификации, основанные на традиционных многоразовых паролях, не обладают достаточной безопасностью. Более надежными являются процедуры аутентификации на основе одноразовых паролей OTP (One Time Password).

Суть схемы одноразовых паролей – использование различных паролей при каждом новом запросе на предоставление доступа. Одноразовый динамический пароль действителен только для одного входа в систему, и затем его действие истекает. Даже если кто-то перехватил его, пароль окажется бесполезным. Динамический механизм задания пароля является одним из лучших способов защитить процесс аутентификации от угроз извне.

Одноразовые пароли генерируются с помощью ОТР-токена. Для этого используется секретный ключ пользователя, размещенный как внутри OTP-токена, так и на сервере аутентификации.

Для того чтобы получить доступ к необходимым ресурсам, пользователь должен ввести пароль, созданный с помощью OTP-токена. Этот пароль сравнивается со значением, сгенерированным на сервере аутентификации, после чего выносится решение о предоставлении доступа. Преимуществом такого подхода является то, что пользователю не требуется соединять токен с компьютером (в отличие от вышеперечисленных типов идентификаторов).

Однако количество приложений ИТ-безопасности, которые поддерживают возможность работы с OTP-токенами, намного меньше, чем для смарт-карт и USB-токенов. Недостатком OTP-токенов является ограниченное время жизни этих устройств (три-четыре года), так как автономность работы предполагает использование батарейки.

Обычно системы аутентификации с одноразовыми паролями используются для проверки удаленных пользователей.

**Реализация.**

Технологии OTP были разработаны в рамках отраслевой инициативы Open Authentication (OATH), выдвинутой компанией VeriSign в 2004 г. Суть этой инициативы заключается в разработке стандартной спецификации действительно надежной аутентификации для различных Интернет-сервисов. Причем речь идет о двухфакторном определении прав пользователя, в процессе которого последний должен "предъявить" смарт-карту или USB-токен и свой пароль. Таким образом, одноразовые пароли со временем могут стать стандартным средством удаленной аутентификации в различных системах.

Сегодня разработано и используется на практике несколько вариантов реализации систем аутентификации по одноразовым паролям.

**Метод "запрос-ответ".** Принцип его работы таков: в начале процедуры аутентификации пользователь отправляет на сервер свой логин. В ответ на это последний генерирует некую случайную строку и посылает ее обратно. Пользователь с помощью своего ключа зашифровывает эти данные и возвращает их серверу. Сервер же в это время "находит" в своей памяти секретный ключ данного пользователя и кодирует с его помощью исходную строку. Далее проводится сравнение обоих результатов шифрования. При их полном совпадении считается, что аутентификация прошла успешно. Этот метод реализации технологии одноразовых паролей называется асинхронным, поскольку процесс аутентификации не зависит от истории работы пользователя с сервером и других факторов.

**Метод "только ответ".** В этом случае алгоритм аутентификации несколько проще. В самом начале процесса программное или аппаратное обеспечение пользователя самостоятельно генерирует исходные данные, которые будут зашифрованы и отправлены на сервер для сравнения. При этом в процессе создания строки используется значение предыдущего запроса. Сервер тоже обладает этими сведениями; зная имя пользователя, он находит значение предыдущего его запроса и генерирует по тому же алгоритму точно такую же строку. Зашифровав ее с помощью секретного ключа пользователя (он также хранится на сервере), сервер получает значение, которое должно полностью совпадать с присланными пользователем данными.

**Метод "синхронизация по времени".** В нем в качестве исходной строки выступают текущие показания таймера специального устройства или компьютера, на котором работает человек. При этом обычно используется не точное указание времени, а текущий интервал с установленными заранее границами (например, 30 с). Эти данные зашифровываются с помощью секретного ключа и в открытом виде отправляются на сервер вместе с именем пользователя. Сервер при получении запроса на аутентификацию выполняет те же действия: получает текущее время от своего таймера и зашифровывает его. После этого ему остается только сравнить два значения: вычисленное и полученное от удаленного компьютера.

Метод "синхронизация по событию". В принципе этот метод практически идентичен предыдущему, только в качестве исходной строки в нем используется не время, а количество успешных процедур аутентификации, проведенных до текущей. Это значение подсчитывается обеими сторонами отдельно друг от друга.

В некоторых системах реализуются так называемые смешанные методы, где в качестве начального значения используется два типа информации или даже больше. Например, существуют системы, которые учитывают, как счетчики аутентификаций, так и показания встроенных таймеров. Такой подход позволяет избежать множества недостатков отдельных методов.

**Уязвимости технологий OTP.**

Технология одноразовых паролей считается достаточно надежной. Однако объективности ради отметим, что и у нее есть недостатки, которым подвержены все системы, реализующие принцип OTP в чистом виде. Подобные уязвимости можно разделить на две группы. К первой относятся потенциально опасные "дыры", присущие всем методам реализации. Наиболее серьезная из них - возможность подмены сервера аутентификации. При этом пользователь будет отправлять свои данные прямо злоумышленнику, который может тут же использовать их для доступа к настоящему серверу. В случае метода "запрос-ответ" алгоритм атаки немного усложняется (компьютер хакера должен сыграть роль "посредника", пропуская через себя процесс обмена информацией между сервером и клиентом). Впрочем, стоит отметить, что на практике осуществить такую атаку совсем не просто.

Другая уязвимость присуща только синхронным методам и связана с тем, что существует риск рассинхронизации информации на сервере и в программном или аппаратном обеспечении пользователя. Допустим, в какой-то системе начальными данными служат показания внутренних таймеров, и по каким-то причинам они перестают совпадать друг с другом. В этом случае все попытки пользователей пройти аутентификацию будут неудачными (ошибка первого рода). К счастью, в подобных случаях ошибка второго рода (допуск "чужого") возникнуть не может. Впрочем, вероятность возникновения описанной ситуации также крайне мала.

Некоторые атаки применимы только к отдельным способам реализации технологии одноразовых паролей. Для примера опять возьмем метод синхронизации по таймеру. Как мы уже говорили, время в нем учитывается не с точностью до секунды, а в пределах какого-то установленного заранее интервала. Это делается с учетом возможности рассинхронизации таймеров, а также появления задержек в передаче данных. И именно этим моментом теоретически может воспользоваться злоумышленник для получения несанкционированного доступа к удаленной системе. Для начала хакер "прослушивает" сетевой трафик от пользователя к серверу аутентификации и перехватывает отправленные "жертвой" логин и одноразовый пароль. Затем он тут же блокирует его компьютер (перегружает его, обрывает связь и т. п.) и отправляет авторизационные данные уже от себя. И если злоумышленник успеет сделать это так быстро, чтобы интервал аутентификации не успел смениться, то сервер признает его как зарегистрированного пользователя.

Понятно, что для такой атаки злоумышленник должен иметь возможность прослушивания трафика, а также быстрого блокирования компьютера клиента, а это задача не из легких. Проще всего соблюсти эти условия тогда, когда атака задумывается заранее, причем для подключения к удаленной системе "жертва" будет использовать компьютер из чужой локальной сети. В этом случае хакер может заранее "поработать" над одним из ПК, получив возможность управлять им с другой машины. Защититься от такой атаки можно только путем использования "доверенных" рабочих машин (например, собственный ноутбук или КПК) и "независимых" защищенных (например, с помощью SSL) каналов выхода в Интернет.

**Качество реализации.** Надежность любой системы безопасности в значительной степени зависит от качества ее реализации. У всех практических решений есть свои недостатки, которые злоумышленники могут использовать в своих целях, причем эти "дыры" зачастую не имеют прямого отношения к реализуемой технологии. В полной мере это правило применимо и к системам аутентификации на базе одноразовых паролей. Как уже говорилось выше, в их основе лежит использование криптографических алгоритмов. Это накладывает определенные обязательства на разработчиков таких продуктов - ведь некачественное исполнение какого-либо алгоритма или, например, генератора случайных чисел может поставить под угрозу безопасность информации.

Генераторы одноразовых паролей реализуются двумя способами: программным и аппаратным. Первый из них, естественно, менее надежен. Дело в том, что клиентская утилита должна хранить в себе секретный ключ пользователя. Сделать это более или менее безопасно можно только с помощью шифрования самого ключа на основе персонального пароля. При этом необходимо учитывать, что клиентская утилита должна быть установлена на том устройстве (КПК, смартфон и т. п.), с которого в данный момент выполняется сессия. Таким образом, получается, что аутентификация сотрудника зависит от одного пароля, при том что существует множество способов узнать или подобрать его. И это далеко не единственная уязвимость программного генератора одноразовых паролей.

Несравнимо большей надежностью обладают разнообразные устройства для аппаратной реализации OTP-технологий. Например, есть устройства, по виду напоминающие калькулятор (рис. 2.1.): при вводе в них набора цифр, присланного сервером, они на основе вшитого секретного ключа генерируют одноразовый пароль (метод "запрос-ответ"). Главная уязвимость подобных устройств связана с тем, что их можно украсть или утерять. Обезопасить систему от злоумышленника можно только при условии использования надежной защиты памяти устройства с секретным ключом.

**2.3. Аутентификация на основе PIN-кода**

Наиболее распространенным методом аутентификации держателя пластиковой карты и смарт-карты является ввод секретного числа, которое обычно называют PIN-кодом (Personal Identification Number — персональный идентификационный код) или иногда CHV (CardHolder Verification). Защита PIN-кода карты является критичной для безопасности всей системы. Карты могут быть потеряны, украдены или подделаны. В таких случаях единственной контрмерой против несанкционированного доступа остается секретное значение PIN-кода. Вот почему открытая форма PIN должна быть известна только законному держателю карты. Очевидно, значение PIN нужно держать в секрете в течение всего срока действия карты.

Длина PIN-кода должна быть достаточно большой, чтобы минимизировать вероятность определения правильного PIN-кода методом проб и ошибок. С другой стороны, длина PIN-кода должна быть достаточно короткой, чтобы дать возможность держателям карт запомнить его значение. Согласно рекомендации стандарта ISO 9564-1, PIN-код должен содержать от 4 до 12 буквенно-цифровых символов. Однако в большинстве случаев ввод нецифровых символов технически невозможен, поскольку доступна только цифровая клавиатура. Поэтому обычно PIN-код представляет собой четырехразрядное число, каждая цифра которого может принимать значение от 0 до 9.

PIN-код вводится с помощью клавиатуры терминала или компьютера и затем отправляется на смарт-карту. Смарт-карта сравнивает полученное значение PIN-кода с эталонным значением, хранимым в карте, и отправляет результат сравнения на терминал. Ввод PIN-кода относится к мерам безопасности, особенно для финансовых транзакций, и, следовательно, требования к клавиатуре часто определяются в прикладной области. PIN-клавиатуры имеют все признаки модуля безопасности и шифруют PIN-код сразу при его вводе. Это обеспечивает надежную защиту от проникновения в клавиатуру для перехвата PIN-кода во время ввода.

При идентификации клиента по значению PIN-кода и предъявленной карте используются два основных способа проверки PIN-кода: неалгоритмический и алгоритмический.

Неалгоритмический способ проверки PIN-кода не требует применения специальных алгоритмов. Проверка PIN-кода осуществляется путем непосредственного сравнения введенного клиентом PIN-кода со значениями, хранимыми в БД. Обычно БД со значениями PIN-кодов клиентов шифруется методом прозрачного шифрования, чтобы повысить ее защищенность, не усложняя процесса сравнения.

Алгоритмический способ проверки PIN-кода заключается в том, что введенный клиентом PIN-код преобразуют по определенному алгоритму с использованием секретного ключа и затем сравнивают со значением PIN-кода, хранящимся в определенной форме на карте. Достоинства этого метода проверки:

отсутствие копии PIN-кода на главном компьютере исключает его раскрытие обслуживающим персоналом;

отсутствие передачи PIN-кода между банкоматом или кас- сиром-автоматом и главным компьютером банка исключает его перехват злоумышленником или навязывание результатов сравнения;

упрощение работы по созданию программного обеспечения системы, так как уже нет необходимости действий в реальном масштабе времени.

# 3.Строгая аутентификация

## 3.1.Основные понятия

Идея строгой аутентификации, реализуемая в криптографических протоколах, заключается в следующем. Проверяемая (доказывающая) сторона доказывает свою подлинность проверяющей стороне, демонстрируя знание некоторого секрета. Например, этот секрет может быть предварительно распределен безопасным способом между сторонами аутентификационного обмена. Доказательство знания секрета осуществляется с помощью последовательности запросов и ответов с использованием криптографических методов и средств.

Существенным является факт, что доказывающая сторона демонстрирует только знание секрета, но сам секрет в ходе аутентификационного обмена не раскрывается. Это обеспечивается посредством ответов доказывающей стороны на различные запросы проверяющей стороны. При этом результирующий запрос зависит только от пользовательского секрета и начального запроса, который обычно представляет произвольно выбранное в начале протокола большое число.

В большинстве случаев строгая аутентификация заключается в том, что каждый пользователь аутентифицируется по признаку владения своим секретным ключом. Иначе говоря, пользователь имеет возможность определить, владеет ли его партнер по связи надлежащим секретным ключом и может ли он использовать этот ключ для подтверждения того, что он действительно является подлинным партнером по информационному обмену.

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

* односторонняя аутентификация;
* двусторонняя аутентификация;
* трехсторонняя аутентификация.

*Односторонняя аутентификация* предусматривает обмен информацией только в одном направлении.

*Двусторонняя аутентификация* по сравнению с односторонней содержит дополнительный ответ проверяющей стороны доказывающей стороне, который должен убедить ее, что связь устанавливается именно с той стороной, которой были предназначены аутентификационные данные;

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

Следует отметить, что данная классификация достаточно условна. На практике набор используемых приемов и средств зависит непосредственно от конкретных условий реализации процесса аутентификации. Необходимо учитывать, что проведение строгой аутентификации требует обязательного согласования сторонами используемых криптографических алгоритмов и дополнительных параметров.

Прежде чем перейти к рассмотрению конкретных вариантов протоколов строгой аутентификации, следует остановиться на назначении и возможностях так называемых одноразовых параметров, используемых в протоколах аутентификации. Одноразовые параметры иногда называют также *nonces* — это величина, используемая для одной и той же цели не более одного раза. Среди используемых на сегодняшний день одноразовых параметров следует выделить: случайные числа, метки времени и номера последовательностей.

Одноразовые параметры позволяют избежать повтора передачи, подмены стороны аутентификационного обмена и атаки с выбором открытого текста. С их помощью можно обеспечить уникальность, однозначность и временные гарантии передаваемых сообщений. Различные типы одноразовых параметров могут употребляться как отдельно, так и дополнять друг друга.

Следует отметить, что одноразовые параметры широко используются и в других вариантах криптографических протоколов (например, в протоколах распределения ключевой информации).

В зависимости от используемых криптографических алгоритмов протоколы строгой аутентификации делятся на протоколы, основанные:

на симметричных алгоритмах шифрования;

однонаправленных ключевых хэш-функциях;

асимметричных алгоритмах шифрования;

алгоритмах электронной цифровой подписи.

## 3.2.Строгая аутентификация, основанная на симметричных алгоритмах

Для работы протоколов аутентификации, построенных на основе симметричных алгоритмов, 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 распределенных системах, применяющих технологию симметричного шифрования, часто используются протоколы аутентификации с участием доверенного сервера, с которым каждая сторона разделяет знание ключа. Такой сервер распределяет сеансовые ключи для каждой пары пользователей всякий раз, когда один из них запрашивает аутентификацию другого. Кажущаяся простота данного подхода является обманчивой, на самом деле разработка протоколов аутентификации этого типа является сложной и с точки зрения безопасности не очевидной.

**Протоколы аутентификации с симметричными алгоритмами шифрования.**

Ниже приводятся три примера протоколов аутентификации, специфицированных в ISO/IEC 9798-2. Эти протоколы предполагают предварительное распределение разделяемых секретных ключей.

Рассмотрим следующие варианты аутентификации:

* односторонняя аутентификация с использованием меток времени;
* односторонняя аутентификация с использованием случайных чисел;
* двусторонняя аутентификация.

В каждом из этих случаев пользователь доказывает свою подлинность, демонстрируя знание секретного ключа, так как производит расшифровывание запросов с помощью этого секретного ключа.

При использовании в процессе аутентификации симметричного шифрования необходимо также реализовать механизмы обеспечения целостности передаваемых данных на основе общепринятых способов.

Введем следующие обозначения:

*rА* — случайное число, сгенерированное участником *А;*

*rB —* случайное число, сгенерированное участником *В*;

*tA* — метка времени, сгенерированная участником *А;*

*Eк —* симметричное шифрование на ключе *К* (ключ *K* должен быть предварительно распределен между *А* и *В).*

1. Односторонняя аутентификация, основанная на метках времени:

 $A\rightarrow B:E\_{K}\left(t\_{A},B\right)$. (1)

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

1. Односторонняя аутентификация, основанная на использовании случайных чисел:

 $A\leftarrow B:r\_{B}$; (1)

 $A\rightarrow B: E\_{K}\left(r\_{B},B\right)$*.* (2)

Участник *В* отправляет участнику *А* случайное число *rB.* Участник *А* шифрует сообщение, состоящее из полученного числа *rB* и идентификатора *В,* и отправляет зашифрованное сообщение участнику *В.* Участник *В* расшифровывает полученное сообщение и сравнивает случайное число, содержащееся в сообщении, с тем, которое он послал участнику *А.* Дополнительно он проверяет имя, указанное в сообщении.

1. Двусторонняя аутентификация, использующая случайные значения:

 $A\leftarrow B: r\_{B}$; (1)

 $A\rightarrow B: E\_{K}\left(r\_{A},r\_{B},B\right)$*.* (2)

 $A\leftarrow B: E\_{K}\left(r\_{A},r\_{B}\right)$*.* (3)

При получении сообщения (2) участник *В* выполняет те же проверки, что и в предыдущем протоколе, и дополнительно расшифровывает случайное число *rA* для включения его в сообщение (3) для участника *А.* Сообщение (3), полученное участником *А,* позволяет ему убедиться на основе проверки значений *rA* и *rB,* что он имеет дело именно с участником *В.*

Широко известными представителями протоколов, обеспечивающих аутентификацию пользователей с привлечением в процессе аутентификации третьей стороны, являются протокол распределения секретных ключей Нидхэма и Шредера и протокол Kerberos.

**Протоколы, 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, т. е. не сопровождается обратным преобразованием — расшифровыванием на приемной стороне. Обе стороны (отправитель и получатель) используют одну и ту же процедуру одностороннего шифрования.

Односторонняя хэш-функция *hK*(•) с параметром-ключом *К*, примененная к шифруемым данным *М*, дает в результате хэш-значение *m* (дайджест), состоящее из фиксированного небольшого числа байт (рис. 3). Дайджест *m* = *hK*(*M*) передается получателю вместе с исходным сообщением *М.* Получатель сообщения, зная, какая односторонняя хэш-функция была применена для получения дайджеста, заново вычисляет ее, используя расшифрованное сообщение *М.* Если значения полученного дайджеста *т* и вычисленного дайджеста *т’* совпадают, значит содержимое сообщения *М* не было подвергнуто никаким изменениям.



Рис.3. Применение для аутентификации односторонней хэш-функции с параметром-ключом

Знание дайджеста не дает возможности восстановить исходное сообщение, но позволяет проверить целостность данных. Дайджест можно рассматривать как своего рода контрольную сумму для исходного сообщения. Однако между дайджестом и обычной контрольной суммой имеется и 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 К, который известен только отправителю и получателю, любая модификация исходного сообщения будет немедленно обнаружена.

На рис. 4 показан другой вариант использования односторонней хэш-функции для проверки целостности данных. В этом случае односторонняя хэш-функция *h*(•) не имеет параметра-ключа, но применяется не просто к сообщению М, а к сообщению, дополненному секретным ключом К, т. е. отправитель вычисляет дайджест т = h(M, К). Получатель, извлекая исходное сообщение М, также дополняет его тем же известным ему секретным ключом К, после чего применяет к полученным данным одностороннюю хэш-функцию *h*(•). Результат вычислений — дайджест т’ — сравнивается с полученным по сети дайджестом т.

Рис. 4 – Применение односторонней хэш-функции к сообщению, дополненному секретным ключом К

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

* функция симметричного шифрования *ЕК* заменяется функцией *hK*;
* проверяющий вместо установления факта совпадения полей в расшифрованных сообщениях с предполагаемыми значениями вычисляет значение однонаправленной функции и сравнивает его с полученным от другого участника обмена информацией;
* для обеспечения независимого вычисления значения однонаправленной функции получателем сообщения в протоколе 1 метка времени *tА* должна передаваться дополнительно в открытом виде, а в сообщении (2) протокола 3 случайное число *rА* должно передаваться дополнительно в открытом виде.

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

 $A\leftarrow B: r\_{B}$; (1)

 $A\rightarrow B: r\_{A},h\_{K}\left(r\_{A},r\_{B},B\right)$*.* (2)

 $A\leftarrow B: h\_{K}\left(r\_{A},r\_{B},A\right)$*.* (3)

Заметим, что в сообщение (3) протокола включено поле *А.* Результирующий протокол обеспечивает взаимную аутентификацию и известен как протокол SKID 3.

## 3.3.Строгая аутентификация, основанная на асимметричных алгоритмах

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. Важно отметить, 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, выдавая себя за проверяющего и действуя от его имени.

Аутентификация с использованием асимметричных алгоритмов шифрования

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

 $A\leftarrow B:h\left(r\right),B,P\_{A}\left(r,B\right);$(1)

 $A\rightarrow B: r$*.* (2)

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

В качестве другого примера приведем модифицированный протокол Нидхэма и Шредера, основанный на асимметричном шифровании.

Рассматривая вариант протокола Нидхэма и Шредера, используемый только для аутентификации, будем подразумевать под РB алгоритм шифрования открытым ключом участника В. Протокол имеет следующую структуру:

 $A\rightarrow B: P\_{B}\left(r\_{1},A\right)$*.* (1)

 $A\leftarrow B: P\_{A}\left(r\_{2},r\_{1}\right)$*.* (2)

 $A\leftarrow B: r\_{2}$; (3)

Аутентификация, основанная на использовании цифровой подписи

В рекомендациях стандарта Х.509 специфицирована схема аутентификации, основанная на использовании цифровой подписи, меток времени и случайных чисел.

Для описания этой схемы аутентификации введем следующие обозначения:

tA , rА и rв — временная метка и случайные числа соответственно;

SA — подпись, сгенерированная участником А;

SB — подпись, сгенерированная участником *B*;

cert*A* — сертификат открытого ключа участника А;

cert*B* — сертификат открытого ключа участника В.

Если участники имеют аутентичные открытые ключи, полученные друг от друга, то 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:

 $A\rightarrow B: cert\_{A}, t\_{A}, B, S\_{A}\left(t\_{A},B\right)$ (1)

После принятия данного сообщения участник В проверяет правильность метки времени tA, полученный идентификатор В и, используя открытый ключ из сертификата cert*A* корректность цифровой подписи SA(tA,В).

1. Односторонняя аутентификация с использованием случайных чисел:

 $A\leftarrow B: r\_{B}$; (1)

 $A\rightarrow B:cert\_{A}, r\_{A}, B, S\_{A}\left(r\_{A},r\_{B},B\right)$ (2)

Участник В, получив сообщение от участника А, убеждается, что именно он является адресатом сообщения; используя открытый ключ участника А, взятый из сертификата cert*A*, проверяет корректность подписи SA(rA , rB , В) под числом rА, полученным в открытом виде, числом rB , которое было отослано в сообщении (1), и его идентификатором В. Подписанное случайное число *r*А используется для предотвращения атак с выборкой открытого текста.

1. Двусторонняя аутентификация с использованием случайных чисел:

 $A\leftarrow B: r\_{B}$; (1)

 $A\rightarrow B:cert\_{A}, r\_{A}, B, S\_{A}\left(r\_{A},r\_{B},B\right)$ (2)

 $A\leftarrow B:cert\_{B}, r\_{B}, B, S\_{B}\left(r\_{A},r\_{B},A\right)$ (3)

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

**Заключение**

На данной лекции мы изучили принципы управления доступом к информационным ресурсам. Нами были рассмотрены вопросы: «Идентификация и аутентификация пользователей и программ», «Методы аутентификации, использующие пароли и PIN-коды», «Строгая аутентификация».