**Шифрование**

Мы приступаем к рассмотрению *криптографических сервисов* безопасности, точнее, к изложению элементарных сведений, помогающих составить общее *представление* о *компьютерной криптографии* и ее месте в общей архитектуре информационных систем.

*Криптография* необходима для реализации, по крайней мере, трех *сервисов безопасности*:

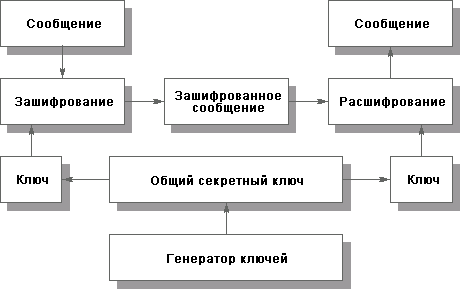
* шифрование;
* контроль целостности;
* аутентификация (этот сервис был рассмотрен нами ранее).

***Шифрование*** – наиболее мощное средство обеспечения конфиденциальности. Во многих отношениях оно занимает центральное место среди программно-технических *регуляторов безопасности*, являясь основой реализации многих из них, и в то же время последним (а подчас и единственным) защитным рубежом. Например, для портативных компьютеров только шифрование позволяет обеспечить конфиденциальность данных даже в случае кражи.

В большинстве случаев и *шифрование*, и *контроль* целостности играют глубоко инфраструктурную роль, оставаясь прозрачными и для приложений, и для пользователей. Типичное *место* этих *сервисов безопасности* – на сетевом и транспортном уровнях реализации стека сетевых протоколов.

Различают два основных метода *шифрования*: *симметричный* и *асимметричный*. В первом из них один и тот же *ключ* (хранящийся в секрете) используется и для зашифрования, и для *расшифрования* данных. Разработаны весьма эффективные (быстрые и надежные) методы симметричного шифрования. Существует и национальный стандарт на подобные методы – ГОСТ 28147-89 "Системы обработки информации. Защита криптографическая. *Алгоритм* криптографического преобразования".

Рис. 11.1 иллюстрирует использование симметричного шифрования. Для определенности мы будем вести речь о защите сообщений, хотя события могут развиваться не только в пространстве, но и во времени, когда зашифровываются и расшифровываются никуда не перемещающиеся файлы.

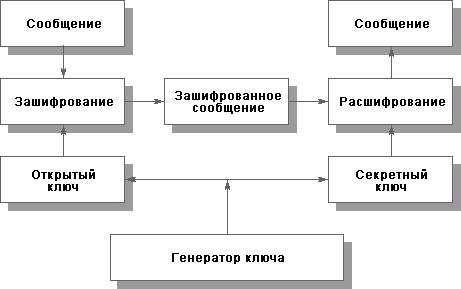


**Рис. 11.1.**Использование симметричного метода шифрования.

Основным недостатком симметричного шифрования является то, что *секретный ключ* должен быть известен и отправителю, и получателю. С одной стороны, это создает новую проблему *распространения ключей*. С другой стороны, получатель на основании наличия зашифрованного и расшифрованного сообщения не может доказать, что он получил это сообщение от конкретного отправителя, поскольку такое же сообщение он мог сгенерировать самостоятельно.

В асимметричных методах используются два ключа. Один из них, несекретный (он может публиковаться вместе с другими открытыми сведениями о пользователе), применяется для шифрования, другой (секретный, известный только получателю) – для расшифрования. Самым популярным из асимметричных является метод *RSA* (Райвест, Шамир, Адлеман), основанный на операциях с большими (скажем, 100-значными) простыми числами и их произведениями.

Проиллюстрируем использование *асимметричного шифрования* (см. рис. 11.2).



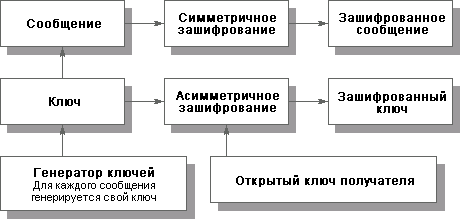
**Рис. 11.2.**Использование асимметричного метода шифрования.

Существенным недостатком асимметричных методов шифрования является их низкое *быстродействие*, поэтому данные методы приходится сочетать с симметричными (асимметричные методы на 3 – 4 порядка медленнее). Так, для решения задачи эффективного шифрования с передачей секретного ключа, использованного *отправителем, сообщение* сначала симметрично зашифровывают случайным ключом, затем этот *ключ* зашифровывают *открытым* асимметричным *ключом* получателя, после чего сообщение и *ключ* отправляются по сети.

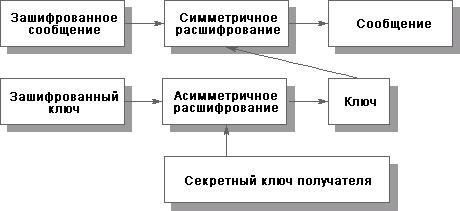
Рис. 11.3 иллюстрирует эффективное *шифрование*, реализованное путем *сочетания* симметричного и асимметричного методов.

На рис. 11.4 показано *расшифрование* эффективно зашифрованного сообщения.

Отметим, что асимметричные методы позволили решить важную задачу совместной выработки секретных ключей (это существенно, если стороны не доверяют друг другу), обслуживающих *сеанс* взаимодействия, при изначальном отсутствии общих секретов. Для этого используется *алгоритм* Диффи-Хелмана.



**Рис. 11.3.**Эффективное шифрование сообщения.



**Рис. 11.4.**Расшифрование эффективно зашифрованного сообщения.

Определенное распространение получила разновидность симметричного шифрования, основанная на использовании *составных ключей*. Идея состоит в том, что секретный *ключ* делится на две части, хранящиеся отдельно. Каждая часть сама по себе не позволяет выполнить*расшифрование*. Если у правоохранительных органов появляются подозрения относительно лица, использующего некоторый *ключ*, они могут в установленном порядке получить половинки ключа и дальше действовать обычным для симметричного расшифрования образом.

Порядок работы с *составными ключами* – хороший пример следования принципу *разделения обязанностей*. Он позволяет сочетать *права* на разного рода тайны (персональную, коммерческую) с возможностью эффективно следить за нарушителями закона, хотя, конечно, здесь очень много тонкостей и технического, и юридического плана.

Многие криптографические алгоритмы в качестве одного из параметров требуют псевдослучайное *значение*, в случае предсказуемости которого в алгоритме появляется *уязвимость* (подобное уязвимое *место* было обнаружено в некоторых вариантах Web-навигаторов). Генерация *псевдослучайных последовательностей* – важный аспект криптографии, на котором мы, однако, останавливаться не будем.

Более подробную информацию о *компьютерной криптографии* можно почерпнуть из статьи Г. Семенова "Не только *шифрование*, или Обзор криптотехнологий" (Jet *Info*, 2001, 3).

### Контроль целостности

Криптографические методы позволяют надежно контролировать *целостность* как отдельных порций данных, так и их наборов (таких как*поток* сообщений); определять подлинность источника данных; гарантировать невозможность отказаться от совершенных действий ("неотказуемость").

В основе криптографического контроля целостности лежат два понятия:

* хэш-функция;
* *электронная цифровая подпись (ЭЦП)*.

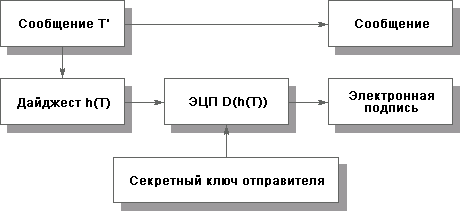
***Хэш-функция*** – это труднообратимое преобразование данных ( *односторонняя функция* ), реализуемое, как правило, средствами симметричного шифрования со связыванием блоков. Результат шифрования последнего блока (зависящий от всех предыдущих) и служит результатом хэш-функции.

Пусть имеются данные, *целостность* которых нужно проверить, хэш-*функция* и ранее вычисленный результат ее применения к исходным данным (так называемый *дайджест* ). Обозначим хэш-функцию через h, исходные данные – через T, проверяемые данные – через T'.*Контроль* целостности данных сводится к проверке равенства h(T') = h(T). Если оно выполнено, считается, что T' = T. Совпадение дайджестов для различных данных называется коллизией. В принципе, коллизии, конечно, возможны, поскольку *мощность множества*дайджестов меньше, чем *мощность множества* хэшируемых данных, однако то, что h есть *функция* односторонняя, означает, что за приемлемое время специально организовать коллизию невозможно.

Рассмотрим теперь применение *асимметричного шифрования* для *выработки* и *проверки электронной цифровой подписи*. Пусть E(T) обозначает результат зашифрования текста T с помощью открытого ключа, а D(T) – результат расшифрования текста Т (как правило, шифрованного) с помощью секретного ключа. Чтобы асимметричный метод мог применяться для реализации *ЭЦП*, необходимо выполнение тождества

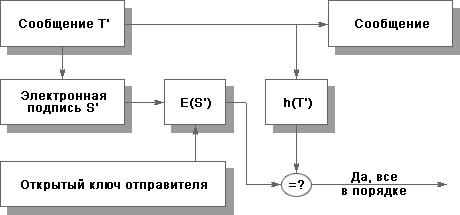
E(D(T)) = D(E(T)) = T

На рис. 11.5 показана процедура выработки электронной цифровой подписи, состоящая в шифровании преобразованием D *дайджеста* h(T).



**Рис. 11.5.**Выработка электронной цифровой подписи.

*Проверка ЭЦП* может быть реализована так, как показано на рис. 11.6.



**Рис. 11.6.**Проверка электронной цифровой подписи.

Из равенства

E(S') = h(T')

следует, что S' = D(h(T')) (для доказательства достаточно применить к обеим частям преобразование D и вычеркнуть в левой части*тождественное преобразование* D(E())). Таким образом, *электронная цифровая подпись* защищает *целостность* сообщения и удостоверяет*личность* отправителя, то есть защищает *целостность* источника данных и служит основой неотказуемости.

Два российских стандарта, ГОСТ Р 34.10-94 "Процедуры выработки и проверки электронной цифровой подписи на базе асимметричного криптографического алгоритма" и ГОСТ Р 34.11-94 "*Функция* хэширования", объединенные общим заголовком "Информационная технология. Криптографическая *защита информации*", регламентируют *вычисление* *дайджеста* и реализацию *ЭЦП*. В сентябре 2001 года был утвержден, а 1 июля 2002 года вступил в силу новый стандарт *ЭЦП* – ГОСТ Р 34.10-2001, разработанный специалистами существовавшего в то время Федерального агентства правительственной связи и информации (ФАПСИ).

Для контроля целостности последовательности сообщений (то есть для защиты от кражи, дублирования и переупорядочения сообщений) применяют временные штампы и нумерацию элементов последовательности, при этом штампы и номера включают в подписываемый текст.

#### Цифровые сертификаты

При использовании асимметричных методов шифрования (и, в частности, электронной цифровой подписи) необходимо иметь гарантию подлинности пары (имя пользователя, открытый ключ пользователя). Для решения этой задачи в спецификациях X.509 вводятся понятия*цифрового сертификата* и *удостоверяющего центра*.

***Удостоверяющий центр*** – это компонент *глобальной службы каталогов*, отвечающий за *управление криптографическими ключами*пользователей.Открытые ключи и другая информация о пользователях хранится удостоверяющими центрами в виде *цифровых сертификатов*, имеющих следующую структуру:

* порядковый номер сертификата;
* идентификатор алгоритма электронной подписи;
* имя *удостоверяющего центра*;
* срок годности;
* имя владельца сертификата (имя пользователя, которому принадлежит сертификат);
* открытые ключи владельца сертификата (ключей может быть несколько);
* идентификаторы алгоритмов, ассоциированных с открытыми ключами владельца сертификата;
* электронная подпись, сгенерированная с использованием секретного ключа *удостоверяющего центра* (подписывается результат хэширования всей информации, хранящейся в сертификате).

Цифровые сертификаты обладают следующими свойствами:

* любой пользователь, знающий открытый ключ *удостоверяющего центра*, может узнать открытые ключи других клиентов центра и проверить целостность сертификата;
* никто, кроме *удостоверяющего центра*, не может модифицировать информацию о пользователе без нарушения целостности сертификата.

В спецификациях X.509 не описывается конкретная процедура генерации криптографических ключей и управления ими, однако даются некоторые общие рекомендации. В частности, оговаривается, что пары ключей могут порождаться любым из следующих способов:

* ключи может генерировать сам пользователь. В таком случае секретный ключ не попадает в руки третьих лиц, однако нужно решать задачу безопасной связи с удостоверяющим центром;
* ключи генерирует доверенное лицо. В таком случае приходится решать задачи безопасной доставки секретного ключа владельцу и предоставления доверенных данных для создания сертификата;
* ключи генерируются удостоверяющим центром. В таком случае остается только задача безопасной передачи ключей владельцу.

Цифровые сертификаты в формате X.509 версии 3 стали не только формальным, но и фактическим стандартом, поддерживаемым многочисленными удостоверяющими центрами.