УНИВЕРСИТЕТ ИТМО

Молдовян А.А., Молдовян Д.Н.,
Левина А.Б.

ПРОТОКОЛЫ АУТЕНТИФИКАЦИИ С НУЛЕВЫМ РАЗГЛАШЕНИЕМ СЕКРЕТА

Санкт-Петербург
2016

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

УНИВЕРСИТЕТ ИТМО

Молдовян А.А., Молдовян Д.Н.,
Левина А.Б.

ПРОТОКОЛЫ АУТЕНТИФИКАЦИИ С НУЛЕВЫМ РАЗГЛАШЕНИЕМ СЕКРЕТА

Учебное пособие

УНИВЕРСИТЕТ ИТМО

Санкт-Петербург

2016

Молдовян А.А., Молдовян Д.Н., Левина А.Б. Протоколы аутентификации с нулевым разглашением секрета.– СПб: Университет ИТМО, 2016. – 55 с.

В данном учебном пособии подробно рассмотрены вопросы аутентификации пользователей информационных систем. Описаны наиболее используемые в настоящее время протоколы аутентификации с нулевым разглашением секрета и представлены математические основы, на которых они базируются.

Данное учебное пособие предназначено для специалистов, осуществляющих подготовку по специальности 090103 «Организация и технология защиты информации», и бакалавров и магистров, осуществляющих подготовку по направлению 090900 «Информационная безопасность».

Рекомендовано к печати Ученым советом факультета КТиУ, 8 декабря 2015 г., протокол №10.

УНИВЕРСИТЕТ ИТМО

Университет ИТМО – ведущий вуз России в области информационных и фотонных технологий, один из немногих российских вузов, получивших в 2009 году статус национального исследовательского университета. С 2013 года Университет ИТМО – участник программы повышения конкурентоспособности российских университетов среди ведущих мировых научно-образовательных центров, известной как проект «5 в 100». Цель Университета ИТМО – становление исследовательского университета мирового уровня, предпринимательского по типу, ориентированного на интернационализацию всех направлений деятельности.

Содержание:

Раздел 1. Основные понятия…….3
1.1 Терминология …….. 3
1.2 Симметричные алгоритмы …….. 5
1.3 Алгоритмы с открытым ключом …….6
1.4 Криптоанализ ……7

Раздел 2. Аутентификация пользователей в информационных системах……9
2.1. Простая аутентификация по паролю……...9
2.2. Протокол рукопожатия…….10

Раздел 3. Протоколы с нулевым разглашением секрета……..13
3.1 Толкование понятия «нулевое разглашение секрета»……13
3.2 Многораундовые протоколы…….15
3.3. Трехшаговые протоколы ……..17

Раздел 4. Двухшаговые протоколы с нулевым разглашением секрета……20
4.1. Протоколы на основе алгоритмов открытого шифрования ……..21
4.2. Протокол на основе использования криптосхемы RSA ……22
4.3. Использование алгоритма открытого шифрования Эль-Гамаля ……..23
4.4. Протокол на основе криптосхемы Рабина……...23

Раздел 5. Протокол на основе схемы открытого согласования ключа…….25
5.1. Реализация с использованием криптосхемы Диффи–Хеллмана……..25
5.2. Реализация с использованием простых чисел Мерсенна…….26

Раздел 6. Протокол на основе последовательного возведения в квадрат……..29
6.1. Протокол на основе последовательного возведения в квадрат……29
6.2. Особенности практического применения……..29

Раздел 7. Преобразование протоколов с нулевым разглашением в схемы цифровой подписи ……31
7.1 Общий подход к выводу схем цифровой подписи и протоколов с нулевым разглашением …….31
7.2 Схема цифровой подписи Фиата–Шамира……32
7.3 Схема цифровой подписи на основе вычислительной трудности задачи дискретного логарифмирования …….34
7.4 Формальное доказательство стойкости схем цифровой подписи ……...37

Раздел 8. Снижение сложности вычислений в криптосхемах, основанных на задаче дискретного логарифмирования ……..40
8.1 Метод Монтгомери ……..40
8.2 Выбор простых чисел специального вида……..42
8.3. Выбор неприводимых многочленов специального вида…….45
8.4. Генерация неприводимых многочленов……47

Раздел 1. Основные понятия

1.1 Терминология

Отправитель и получатель

Одним из основных понятий в криптографии является понятие *отправитель* и *получатель*. Предположим, что отправитель хочет послать сообщение получателю, более того, этот отправитель хочет послать свое сообщение безопасно: он хочет быть уверен, что перехвативший это сообщение не сможет его прочесть. Стороны, обменивающиеся шифрованной информацией, обычно обозначаются *A* и *B*. Довольно часто употребляют более дружественные имена: Алиса и Боб. Но не следует думать, что стороны, участвующие в процессе, обязательно люди. Используя эти имена, мы вполне можем описывать обмен секретной информацией между двумя автономными механизмами. Подслушивающей стороне, «плохой девочке», которая взламывает шифртекст, обычно дают имя Ева.

Сообщения и шифрование

Само сообщение называется **открытым текстом.** Алгоритм **шифрования (или шифр)** — это перевод открытого текста в текст зашифрованный (или шифртекст, шифрограмму, криптограмму) с помощью секретного ключа. Этот процесс называют шифрованием. Обозначим открытый текст как *M* (от *message,* сообщение), или *P* (от *plaintext,* открытый текст). Это может быть поток битов, текстовый файл, битовое изображение, оцифрованный звук, цифровое видеоизображение. Для компьютера *M* — это просто двоичные данные. Открытый текст может быть создан для хранения или передачи, в любом случае, *M* - это сообщение, которое должно быть зашифровано.

Обозначим шифртекст как *C* (от *ciphertext),* это тоже двоичные данные, иногда того же размера, что и *M*, иногда больше. Если шифрование сопровождается сжатием, *C* может быть меньше чем *M.* Однако, само шифрование не обеспечивает сжатие информации. Функция шифрования *E* действует на *M*, создавая *C.*

Мы будем писать

$$C = E_K(M),$$

где *M* — открытый текст, *E* — шифрующая *функция, K* — секретный ключ *и* C — шифртекст.

Обратный процесс называют ***расшифрованием*** и пишут

$$M = D_k(C).$$

Заметим, что алгоритмы шифрования и расшифрования *E* и *D* открыты, и секретность исходного текста *M*, содержащегося в данном шифртексте *C*, обеспечивается секретностью ключа *K*.

Поскольку смыслом шифрования и последующего дешифрирования сообщения является восстановление первоначального открытого текста, должно выполняться следующее равенство:

$$D_k(E_k(M)) = M$$

Искусство и наука безопасных сообщений, называемая **криптографией,** воплощается в жизнь **криптографами. Криптоаналитиками** называются те, кто используют **криптоанализ,** искусство и науку взламывать шифртекст, то есть, раскрывать, то, что было зашифровано. Отрасль математики, охватывающая криптографию и криптоанализ, называется **криптологией**, а люди, которые ей

занимаются – **криптологами.** Для создания криптосистемы, криптограф должен быть хорошим криптоаналитиком.

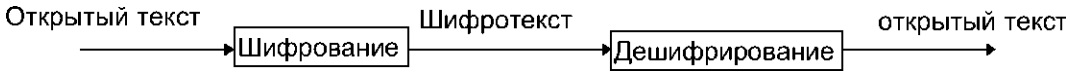

Рис. 1-1. Шифрование и дешифрирование

Проверка подлинности, целостность и неотрицание авторства

Кроме обеспечения конфиденциальности криптография часто используется для других функций:

— **Проверка подлинности.** Получатель сообщения может проверить его источник, злоумышленник не сможет замаскироваться под кого-либо.

— **Целостность.** Получатель сообщения может проверить, не было ли сообщение изменено в процессе доставки, злоумышленник не сможет подменить правильное сообщение ложным.

— **Неотрицание авторства.** Отправитель не сможет ложно отрицать отправку сообщения.

Существуют жизненно важные требования к общению при помощи компьютеров, также как существуют аналогичные требования при общении лицом к лицу. Как раз это и является необходимость обеспечения проверки подлинности, целостности и неотрицания авторства.

Алгоритмы и ключи

Криптографический алгоритм, также называемый **шифром,** представляет собой математическую функцию, используемую для шифрования и дешифрирования. Обычно это две связанных функции: одна для шифрования, а другая для дешифрирования.

Если безопасность алгоритма основана на сохранении самого алгоритма в тайне, это **ограниченный** алгоритм. Ограниченные алгоритмы представляют только исторический интерес, но они совершенно не соответствуют сегодняшним стандартам. Большая или изменяющаяся группа пользователей не может использовать такие алгоритмы, так как всякий раз, когда пользователь покидает группу, ее члены должны переходить на другой алгоритм. Алгоритм должен быть заменен и, если кто-нибудь извне случайно узнает секрет.

Ограниченные алгоритмы не допускают качественного контроля или стандартизации. У каждой группы пользователей должен быть свой уникальный алгоритм. Такие группы не могут использовать открытые аппаратные или программные продукты, поскольку злоумышленник может купить такой же продукт и раскрыть алгоритм. Им приходится разрабатывать и реализовывать собственные алгоритмы.

Несмотря на эти основные недостатки ограниченные алгоритмы необычайно популярны для приложений с низким уровнем безопасности. Обычно пользователи либо не понимают проблем, связанных с безопасностью своих систем, либо не заботятся о них.

Современная криптография решает эти проблемы с помощью **ключа** *K*. Такой ключ может быть любым значением, выбранным из большого множества. Множество возможных ключей называют **пространством ключей.** И шифрование, и дешифрирование использует этот ключ, то есть, они зависят от ключа, что обозначается индексом *K*, и теперь эти функции выглядят как:

$$E_K(M)=C$$

$$D_K(C)=M$$

При этом выполняется следующее равенство:

$$D_K(E_K(M))=M$$

Для некоторых алгоритмов при шифровании и дешифрировании используются различные ключи. То есть ключ шифрования, K_1, отличается от соответствующего ключа дешифрирования, K_2. В этом случае:

$$E_{K1}(M)=C$$

$$D_{K2}(C)=M$$

$$D_{K2}(E_{K1}(M))=M$$

Безопасность этих алгоритмов полностью основана на секретности ключей, а не на том, что потенциальный нарушитель не знает алгоритма. Это значит, что алгоритм может быть опубликован и проанализирован. Продукты, использующие такие алгоритмы, могут широко тиражироваться. Не имеет значения, что злоумышленнику известен алгоритм, поскольку, если ему не известен конкретный ключ, то он не сможет прочесть сообщения.

Криптосистема представляет собой алгоритм плюс все возможные открытые тексты, шифртексты и ключи.

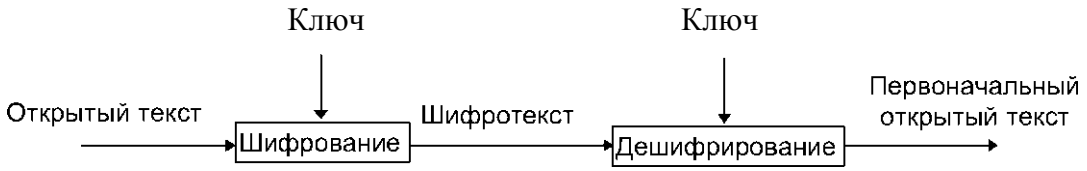

Рис. 1-2. Шифрование и дешифрирование с ключом

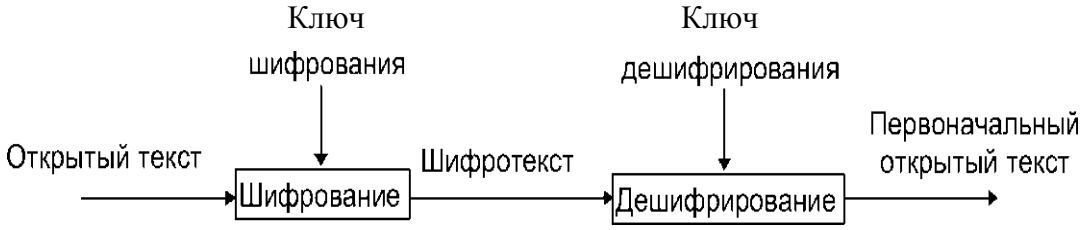

Рис. 1-3. Шифрование и дешифрирование с двумя различными ключами

1.2 Симметричные алгоритмы

Существует два основных типа криптографических алгоритмов: симметричные (с секретным ключом) и асимметричные (с открытым ключом). **Симметричные алгоритмы,** иногда называемые условными алгоритмами, представляют собой алгоритмы, в которых ключ шифрования может быть рассчитан по ключу дешифрирования и наоборот. В большинстве симметричных алгоритмов ключи шифрования и дешифрирования одни и те же. Эти алгоритмы, также называемые алгоритмами с секретным ключом или алгоритмами с одним ключом, требуется, чтобы отправитель и получатель согласовали используемый ключ перед началом безопасной передачи сообщений. Безопасность симметричного алгоритма определяется ключом, раскрытие ключа означает, что кто угодно сможет шифровать и дешифрировать сообщения. Пока передаваемые сообщения должны быть тайными, ключ должен

храниться в секрете. Шифрование и дешифрирование с использованием симметричного алгоритма обозначается как:

$$E_K(M)=C$$

$$D_K(C)=M$$

Симметричные алгоритмы делятся на две категории. Одни алгоритмы обрабатывают открытый текст побитно (иногда побайтно), они называются **потоковыми алгоритмами** или **потоковыми шифрами.** Другие работаю с группами битов открытого текста. Группы битов называются блоками, а алгоритмы – **блочными алгоритмами** или **блочными шифрами.** Для алгоритмов, используемых в компьютерных модемах, типичный размер блока составляет 64 бита - достаточно большое значение, чтобы помешать анализу, и достаточно небольшое и удобное для работы. До появления компьютеров алгоритмы обычно обрабатывали открытый текст посимвольно. Такой вариант может рассматриваться как потоковый алгоритм, обрабатывающий поток символов.

1.3 Алгоритмы с открытым ключом

Алгоритмы с открытым ключом (называемые асимметричными алгоритмами) устроены таким образом, что ключ, используемый для шифрования, отличается от ключа дешифрирования. Более того, ключ дешифрирования не может быть рассчитан по ключу шифрования. Алгоритмы называются "с открытым ключом", потому что ключ шифрования может быть открытым: кто угодно может использовать ключ шифрования для шифрования сообщения, но только конкретный человек, знающий соответствующий ключ дешифрирования, может расшифровать сообщение.

В этих системах ключ шифрования часто называется **открытым** ключом, а ключ дешифрирования – **закрытым.**

Открытый ключ может быть опубликован в справочнике наряду с именем пользователя. В результате любой желающий может зашифровать с его помощью свое письмо и послать закрытую информацию владельцу соответствующего секретного ключа. Расшифровать посланное сообщение сможет только тот, у кого есть секретный ключ. Более точно, имеют место преобразования:

сообщение + ОТКРЫТЫЙ КЛЮЧ АЛИСЫ = ШИФРТЕКСТ

ШИФРТЕКСТ + секретный ключ Алисы = сообщение.

Таким образом, каждый может послать Алисе секретную информацию, воспользовавшись ее открытым ключом. Но только Алиса в состоянии расшифровать сообщение, поскольку лишь у нее есть соответствующий секретный ключ.

Шифрование с открытым ключом *K* обозначается как:

$$E_K(M)=C$$

Хотя открытый и закрытый ключи различны, дешифрирование с соответствующим закрытым ключом обозначается как:

$$D_K(C)=M$$

Иногда сообщения шифруются закрытым ключом, а дешифрируются открытым, что используется для цифровой подписи.

1.4 Криптоанализ

Смысл применения криптографического преобразования исходного текста состоит в обеспечении его секретности при передаче по открытым каналам связи, т.е. в

сохранении открытого текста (или ключа, или и того, и другого) в тайне от злоумышленников (также называемых взломщиками, соперниками, противниками, врагами, перехватчиками). Предполагается, что злоумышленники полностью контролируют линии связи между отправителем и получателем.

Криптоанализ – это наука получения открытого текста из шифртекста без знания секретного ключа. Успешно проведенный криптоанализ может раскрыть открытый текст или ключ, он также может обнаружить слабые места в криптосистемах, что в конце концов приведет к предыдущему результату. Раскрытие ключа без выполнения криптоанализа называется **компрометацией ключа.**

Основное предположение криптоанализа, впервые сформулированное в девятнадцатом веке Огюстом Керкгоффс, состоит в том, что безопасность полностью определяется ключом. Шифрующая, так и расшифровывающая функции общеизвестны, и тайна сообщения, при известном шифртексте, зависит только от секретности ключа.

Существует четыре основных типа криптоаналитического вскрытия. Для каждого из них, предполагается, что криптоаналитик обладает всей полнотой знаний об используемом алгоритме шифрования:

1. **Вскрытие с использованием только шифртекста.** У криптоаналитика есть шифртексты нескольких сообщений, зашифрованных одним и тем же алгоритмом шифрования. Задача криптоаналитика состоит в раскрытии открытого текста как можно большего числа сообщений или, что лучше, получении ключа (ключей), использованного для шифрования сообщений, для дешифрировании других сообщений, зашифрованных теми же ключами.

   Дано: $C_1=E_k(P_1)$, $C_2=E_k(P_2)$, . . . $C_i,=E_k(P_i)$.

   Получить: Либо $P_1, P_2, \ldots P_i$; k; либо алгоритм, как получать P_{i+1} из $C_{i+1}=E_k(P_{i+1})$.

2. **Вскрытие с использованием открытого текста.** У криптоаналитика есть доступ не только к шифртекстам нескольких сообщений, но и к открытому тексту этих сообщений. Его задача состоит в получении ключа (или ключей), использованного для шифрования сообщений, для дешифрирования других сообщений, зашифрованных тем же ключом (ключами).

   Дано: $P_1, C_1=E_k(P_1), P_2, C_2=E_k(P_2), \ldots, P_i, C_i,=E_k(P_i)$.

   Получить: Либо k; либо алгоритм, как получать P_{i+1} из $C_{i+1}=E_k(P_{i+1})$

3. **Вскрытие с использованием выбранного открытого текста.** Предполагается, что у криптоаналитика есть не только доступ к шифртекстам и открытым текстам нескольких сообщений, но и возможность выбирать открытый текст для шифрования. Это расширяет способы криптоанализа по сравнению со случаем известного открытого текста, так как криптоаналитик может выбирать шифруемые блоки открытого текста и получать соответствующие им блоки шифртекста. Выполняя такой выбор с учетом особенностей алгоритма шифрования, можно получить больше информации о ключе. Задача криптоаналитика состоит в получении ключа (или ключей), использованного для шифрования сообщений, или алгоритма, позволяющего дешифрировать новые сообщения, зашифрованные тем же ключом (или ключами).

   Дано: $P_1, C_1=E_k(P_1), P_2, C_2=E_k(P_2), \ldots, P_i, C_i,=E_k(P_i)$, где криптоаналитик может выбирать $P_1, P_2, \ldots P_i$

   Получить: Либо k; либо алгоритм, как получать P_{i+1} из $C_{i+1}=E_k(P_{i+1})$

4. **Адаптивное вскрытие с использованием открытого текста.** Это частный случай вскрытия с использованием выбранного открытого текста. Криптоаналитик не только может многократно выбирать шифруемый текст, но также может строить свой последующий выбор на базе полученных результатов шифрования предыдущих выбранных текстов. При вскрытии с использованием выбранного открытого текста криптоаналитик мог выбрать для шифрования только один большой блок открытого текста, при адаптивном вскрытии с использованием выбранного открытого текста он может выбрать меньший блок открытого текста, затем выбрать следующий блок, используя результаты первого выбора и так далее.

5. **Вскрытие с использованием выбранного шифртекста.** Криптоаналитик может выбрать различные шифртексты для дешифрирования и имеет доступ к дешифрированным открытым текстам. Например, у криптоаналитика есть доступ к "черному ящику", который выполняет автоматическое дешифрирование. Его задача состоит в получении ключа.

Дано: $C_1, P_1=D_k(C_1), C_2, P_2=D_k(C_2), \ldots, C_i, P_i=D_k(C_i)$.

Получить: k.

Такой тип вскрытия обычно применим к алгоритмам с открытым ключом. Вскрытие с использование выбранного шифртекста иногда также эффективно против симметричных алгоритмов. Иногда вскрытие с использованием выбранного открытого текста и вскрытие с использованием выбранного шифртекста вместе называют вскрытием с использованием выбранного текста.

6. **Вскрытие с использованием выбранного ключа.** Такой тип вскрытия означает то, что криптоаналитик может выбирать ключ и что у него есть некоторая информация о связи между различными ключами.

7. **Бандитский криптоанализ.** Криптоаналитик угрожает, шантажирует или пытает кого-нибудь, пока не получит ключ. Взяточничество иногда называется **вскрытием с покупкой ключа.** Это очень мощные способы вскрытия, часто являющиеся наилучшим путем взломать алгоритм.

Вскрытия с известным открытым текстом и с использованием выбранного открытого текста встречаются чаще, чем можно подумать. Многие сообщения имеют стандартные начало и окончание, что может быть известно криптоаналитику. Особенно уязвим шифрованный исходный код из-за частого использования ключевых слов: #define, struct, else, return. Те же проблемы и у шифрованного исполнимого кода: функции, циклические структуры и так далее. Вскрытия на основе известного открытого текста (и вскрытия на основе выбранного шифртекста) успешно использовались для чтения немецких и японских секретных сообщений, передаваемых в виде криптограмм по радиоканалу, в ходе Второй мировой войны.

Лучшими алгоритмами являются те, которые были разработаны открыто и анализировались длительное время лучшими криптоаналитиками.

В данном учебном пособии будут подробно рассмотрены протоколы **аутентификации** пользователей. Аутентификация является процедурой проверки подлинности и имеет огромное значение для информационных систем в настоящее время.

Раздел 2. Аутентификация пользователей в информационных системах

2.1. Простая аутентификация по паролю

В современных информационно-телекоммуникационных системах процедуры аутентификации удаленных пользователей весьма широко применяются. Можно выделить следующие типы протоколов аутентификации пользователей:

- *протоколы простой аутентификации* на основе паролей (наиболее уязвимый тип при взаимодействии через канал связи, поскольку пароли представляются в открытом виде без их преобразования, но несмотря на это парольная аутентификация достаточно распространена в Интернет, поскольку дает удобный механизм разграничения доступа к удаленным ресурсам без использования защищенных каналов связи и без привлечения доверенных сторон);

- *протоколы строгой аутентификации* (обеспечивает аутентификацию удаленных пользователей без явного разглашения секретного ключа);

- *протоколы с нулевым разглашением* (протоколы данного типа основанны на предварительном распределении открытых ключей и их аутентификации; каждый из взаимодействующих субъектов убеждается в подлинности другого субъекта с использованием открытого ключа последнего, причем после процедуры аутентификации никакой из субъектов не получает новой информации о секретном ключе субъекта, подлинность которого подтверждена в ходе осуществления протокола; до выполнения протокола общедоступна информация о секретном ключе, которая состоит в том, что секретный ключ связан с открытым ключом, однако его вычисление невыполнимо за обозримое время).

Частным вариантом использования процедуры простой аутентификации является парольная защита входа в компьютерную систему. Например, пользователь формирует некоторую случайную информацию и, сохраняя ее в секрете, использует как пароль. Пароль в явном виде не хранится в памяти ЭВМ (или другого устройства, применяемого для выполнения аутентификации). Это требование направлено на то, чтобы потенциальный внутренний нарушитель не имел возможности извлечь из машинной памяти чужой пароль и присвоить себе полномочия другого пользователя. Для того, чтобы система защиты могла идентифицировать легальных (санкционированных) пользователей, в памяти ЭВМ хранятся образы их паролей, вычисленные по специальному криптографическому алгоритму, реализующему, так называемую, одностороннюю функцию $y = F(x)$, где F криптографический алгоритм - односторонняя функция. Основное требование к односторонней функции состоит в том, чтобы сложность вычисления значения функции по аргументу (по входу) была низкой, а сложность определения значения аргумента, для которого значение функции (значение выхода) было бы равно случайно выбранному значению, была высокой (неосуществимой за обозримое время при использовании всех вычислительных ресурсов доступных предполагаемому нарушителю).

Аутентификация пользователя на рабочей станции может быть выполнена следующим образом:

1. Запрос на ввод идентификатора со стороны системы защиты.
2. Ввод пользователем своего идентификатора (имени) NAME.
3. Запрос на ввод пароля со стороны системы защиты.
4. Ввод пользователем пароля P.
5. Вычисление системой защиты значения односторонней функции y,

соответствующей значению аргумента $x = P$.

6. Сравнение системой защиты значения $F(P)$ со значением образа (S) пароля, соответствующего пользователю с идентификатором NAME.

Если $F(P) = S$, то система защиты предоставляет пользователю права доступа (полномочия), соответствующие идентификатору NAME. В противном случае в журнале учета работы пользователей регистрируется событие попытки несанкционированного доступа. Для того, чтобы выдать себя за санкционированного пользователя нарушитель должен ввести правильный пароль. Зная образ S, вычислительно невозможно определить пароль P. Если в системе защиты предусмотрены механизмы противодействия перехвату пароля с помощью программных или аппаратных закладок, а также через побочные электромагнитные излучения и наводки (ПЭМИН), акустический и оптический канал, то данный способ аутентификации пользователей обеспечивает высокую надежность защиты от узурпирования чужих полномочий. Рассмотренный пример относится к аутентификации пользователей на рабочих станциях, т. е. к задаче защиты входа в ЭВМ.

2.2. Протокол рукопожатия

Данный протокол относится к протоколам строгой аутентификации. В действительности «протокол рукопожатия» – это общее название ряда интерактивных протоколов строгой взаимной аутентификации двух пользователей, в которых используется механизм запроса–ответа, с помощью которого каждая из взаимодействующих сторон убеждается, что другая сторона знает некоторый секрет K. Схемы аутентификации такого типа называются протоколами взаимной аутентификации, или протоколами рукопожатия. В качестве секрета может служить разделяемый сторонами общий секретный ключ или личный секретный ключ, связанный с открытым ключом стороны, подлинность которой проверяется. В первом случае ключ доставляется от одной стороны к другой по защищенному каналу или через доверенное лицо (для этого можно также выполнить протокол открытого согласования ключа, но это требует использования открытых ключей, однако их можно сразу использовать в протоколе рукопожатия). Во втором случае открытые ключи друг друга стороны берут из справочника открытых ключей или цифровых сертификатов, подписанных удостоверяющим центром.

Важным моментом является то, что запрос должен представлять собой случайное сообщение. Это предотвращает использование нарушителем (с целью выдать себя за законного абонента) копий старых сообщений, переданных между пользователями в процессе предыдущей процедуры аутентификации. Рассмотрим некоторые варианты реализации протокола рукопожатия.

2.2.1 Протокол с использованием блочных шифров.

При использовании некоторого 64-битового блочного шифра E и разделяемого секретного ключа K взаимная аутентификация абонентов A и B осуществляется следующим образом.

1. Абонент B генерирует некоторое случайное число R произвольной длины и посылает его абоненту A.

2. Абонент A, используя заранее оговоренный алгоритм блочного шифрования, шифрует полученное случайное число R по ключу K: $C = E_K(R)$. Затем он генерирует свое случайное число R' и пересылает значения R' и C абоненту B.

3. Абонент B самостоятельно осуществляет такое же преобразование числа R: $C^* = E_K(R)$. Если $C^* = C$, то абонент B принимает решение, что абонент A является

подлинным. Затем абонент B шифрует полученное случайное число R' по ключу K: $C' = E_K(R')$ и пересылает значение C' абоненту A.

Получив ответ C' на свой случайный запрос R', абонент A самостоятельно осуществляет преобразование числа R' и получает шифртекст $C'' = E_K(R')$. Если $C'' = C'$, то абонент A принимает решение, что абонент B является подлинным.

2.2.2 Протокол с использованием бесключевых хэш-функций.

Пусть дана некоторая стойкая хэш-функция F_H, абоненты A и B заранее договорились об использовании разделяемого секретного ключа K. Их взаимная аутентификация может быть осуществляется следующим образом.

1. Абонент B генерирует некоторое случайное число R произвольного размера и посылает его абоненту A.

2. Абонент A, используя хэш-функцию F_H вычисляет по ключу K значение $h = F_H(K, R)$. Затем он генерирует свое случайное число R' и пересылает значения R' и h абоненту B.

3. Абонент B самостоятельно вычисляет значение $h^* = F_H(K, R)$. Если $h^* = h$, то абонент B принимает решение, что абонент A является подлинным. Затем абонент B вычисляет значение $h' = F_H(K, R')$ и отправляет значение h' абоненту A.

Получив ответ h' на свой случайный запрос R', абонент A самостоятельно вычисляет значение $h'' = F_H(K, R')$. Если $h'' = h'$, то абонент A принимает решение, что абонент B является подлинным.

2.2.3 Протокол с использованием цифровых подписей.

Пусть абоненты A и B обменялись цифровыми сертификатами, в которых указаны их открытые ключи y_A и y_B, соответственно, и договорились об использовании некоторой схемы цифровой подписи. Их взаимная аутентификация может быть осуществляется следующим образом.

1. Абонент B генерирует некоторое случайное число R произвольной длины и посылает его абоненту A.

2. Абонент A, используя свой секретный ключ вычисляет значение своей цифровой подписи $S_A(R)$ к запросу R. Затем он генерирует свой случайный запрос R' и пересылает значения R' и $S_A(R)$ абоненту B.

3. Абонент B проверяет подлинность подписи $S_A(R)$. Если подпись $S_A(R)$ является подлинной, то абонент B принимает решение о подлинности абонента A. Затем абонент B вычисляет значение своей подписи $S_B(R')$ к R' и отправляет значение $S_B(R')$ абоненту A.

Получив ответ $S_B(R')$ на свой случайный запрос R', абонент A проверяет подлинность подписи $S_B(R')$. Если подпись $S_B(R')$ является подлинной, то абонент A принимает решение о подлинности абонента B.

2.2.4 Протокол с использованием открытого шифрования.

Пусть абоненты A и B обменялись цифровыми сертификатами, в которых указаны их открытые ключи (n_A, e_A) и (n_B, e_B), действующие в рамках криптосхемы RSA [19]. Протокол рукопожатия реализуется следующим образом.

1. Абонент B генерирует некоторое случайное число $R < n_A$, зашифровывает его по открытому ключу (n_A, e_A): $C = R^{e_A} \bmod n_A$. Затем посылает шифртекст C абоненту A.

2. Абонент А, используя свой личный секретный ключ d_A, расшифровывает полученный шифртекст C и получает значение $R^* = C^{d_A} \bmod n_A$. Затем он генерирует свой случайный запрос R', зашифровывает его по открытому ключу (n_B, e_B): $C' = R'^{e_B} \bmod n_A$. и пересылает значения C' и R^* абоненту В.

3. Абонент В проверяет выполнимость равенства $R^* = R$. Если оно верно, то, используя свой секретный ключ d_B, абонент В расшифровывает полученный шифртекст C' и получает значение $R'' = C'^{d_B} \bmod n_B$, которое направляет абоненту А.

Получив ответ R' на свой случайный запрос C', абонент А проверяет выполнимость равенства $R'' = R'$. Если оно верно, то абонент А принимает решение о подлинности абонента В.

Несмотря на общее сходство идеи построения всех четырех описанных вариантов, последний протокол обладает следующей потенциальной уязвимостью, если в модели нарушителя предусматривается возможность того, что нарушитель может пытаться выдать себя за одного из абонентов протокола. Действительно, нарушитель может формировать свой запрос не так как указано в протоколе (где сначала выбирается ожидаемый ответ, а потом путем шифрования последнего вычисляется запрос), а формировать значение запроса произвольно с целью получения информации о личном секрете того по полученному ответу.

Например, при использовании алгоритма открытого шифрования Эль-Гамаля [1,2] с простым модулем p вместо RSA нарушитель может в качестве своего запроса направить пару чисел значение (ρ, C), где ρ – число сравнительно малого порядка по модулю p. Абонент, который должен дать ответ R на этот запрос, в соответствии с протоколом вычисляет R как исходный текст, восстановленный из криптограммы (ρ, C): $R = \rho^{-x} C \bmod p$, где x – секретный ключ абонента. Получив ответ R, нарушитель вычисляет значение $\rho^x = R^{-1} C \bmod p$, по которому он может вычислить значение $x' = x \bmod \omega(\rho)$, где $\omega(\rho)$ – значение порядка числа ρ. Получив несколько ответов при различных значениях ρ, нарушитель сможет по китайской теореме об остатках вычислить значение секретного ключа абонента.

Протокол рукопожатия позволяет установить взаимную подлинность двух взаимодействующих субъектов. Но это не означает, что теперь они могут безопасно обменяться открытыми ключами, зашифровав их на секретном ключе. Дело в том, что они могут в дальнейшем отказаться от своих открытых ключей и третья сторона не сможет выявить, кто из этих двух субъектов является нарушителем. Кроме того, нужно оценить потенциальную атаку, состоящую в том, что нарушитель может направлять ложные сообщения сразу после того как процедура взаимной аутентификации выполнена успешно. Это типичный случай для атак типа «нарушитель посередине». Для противодействия таким атакам следует каждое сообщение сопровождать цифровой подписью отправителя, а получателю – проверять подлинность подписи к каждому полученному сообщению.

Раздел 3. Протоколы с нулевым разглашением секрета

Интересным классом протоколов являются протоколы с нулевым разглашением. Такое условное название они получили ввиду того, что в процессе их реализации один пользователь (доказывающий) может убедить другого (проверяющего) в том, что он владеет некоторым секретом, без раскрытия самого секрета. Вернее сказать, без того, чтобы дать какую-либо дополнительную информацию о секрете к той, которую мог бы получить проверяющий, из имеющихся исходных данных до начала реализации протокола, т. е. в процессе выполнения протокола с нулевым разглашением не происходит какой-либо утечки информации о ключе. Это актуально, так как протоколы такого типа строятся с использованием открытого ключа, который содержит в себе полную информацию о личном секретном ключе владельца. Толкования смысла понятия нулевое разглашения будет рассмотрено далее.

Возможное применение протоколов с нулевым разглашением секрета связано с подтверждением подлинности удаленных субъектов (пользователей). Протоколы с нулевым разглашением являются доказуемо стойкими двухключевыми криптосхемами.

Протоколы данного вида, включающие следующие три типовых шага:

1) генерация фиксатора в виде разового открытого ключа доказывающим;

2) генерация случайного запроса проверяющим;

3) вычисление доказывающим ответа на запрос.

Данные протоколы имеют и другое важное значение, оно состоит в возможности их преобразования в схемы электронной цифровой подписи (ЭЦП).

В протоколах с нулевым разглашением используются вычислительно трудные задачи следующих типов:

1) задача дискретного логарифмирования в конечной циклической группе (или в конечном поле);

2) задача факторизации;

3) задача извлечения корней по модулю большого простого числа;

4) задача о раскраске графа и некоторые другие.

3.1 Толкование понятия «нулевое разглашение секрета»

При изучении протоколов с нулевыми знаниями важным методическим аспектом является разъяснение термина «нулевое разглашение секрета». С понятием нулевого разглашения секрета можно связать понятие стойкости протокола к атакам различного типа. Стойкость криптографических схем (алгоритмов и протоколов) является одним из центральных понятий в криптографии. При этом для симметричных криптосистем различают условную (практическую) и безусловную (теоретическую) стойкость. Практическая стойкость связана с тем, что имеющиеся в настоящее время вычислительные ресурсы (уровень развития вычислительной техники) не позволяют за обозримое время взломать криптосистему, тогда как это принципиально возможно. Теоретическая стойкость предполагает стойкость криптосистемы при наличии у атакующего бесконечных вычислительных ресурсов. По своей природе двухключевые криптосхема не могут обладать теоретической стойкостью и относятся к классу практически стойких криптосхем. Для двухключевых криптосхем используется понятие доказуемой стойкости. Двухключевая криптосхема называется доказуемо стойкой, если можно формально доказать, что взлом криптосхемы не проще, чем решение вычислительно трудной задачи, лежащей в ее основе.

К доказуемо стойким криптосхемам можно отнести и протоколы с нулевым разглашением секрета, в основе которых лежат вычислительно трудные задачи, используемые для вычисления открытого ключа по секретному ключу. Благодаря практической невыполнимости обратной задачи – вычисления секретного ключа по открытому, открытый ключ может быть сделан общедоступным параметром протокола. Таким образом, у потенциального нарушителя имеется информация, дающая ему принципиальную возможность вычислить секретный ключ. Этой информацией является открытый ключ, но вычислить из него секретный ключ – нереализуемая задача. Поэтому, несмотря на теоретическую полноту этой информации, нарушитель не имеет практической возможности ею воспользоваться. Если в ходе протокола аутентификации к указанной информации не добавляется какая-либо новая информация, то говорят о нулевой утечке информации о секретном ключе или нулевом разглашении секрета.

Рассмотрим вопрос о том, когда можно считать, что владелец открытого ключа, по которому выполняется процедура его аутентификации, в ходе выполнения протокола не передает никаких знаний о секрете проверяющей стороне. Очевидно, что владелец открытого ключа (доказывающий) неминуемо должен воспользоваться своим секретным ключом. Однако, протокол позволяет проверяющему субъекту удостовериться с вероятностью сколь угодно близкой к единице, что аутентифицируемый субъект действительно знает секретный ключ. То, что передаваемые доказывающим значения не содержат в себе информации о секретном ключе, понимается в том смысле, что эти значения не упрощают для потенциального нарушителя задачи вычисления секретного ключа по связанному с ним открытому ключу. Это имеет место в следующих случаях.

- Доказывающий передает проверяющему значение, которое заведомо известно последнему.
- Доказывающий передает проверяющему значение, которое последний может вычислить самостоятельно до получения ответа.
- Доказывающий передает проверяющему случайные значения.
- Доказывающий передает проверяющему случайные пары или наборы случайных значений, удовлетворяющих некоторому проверочному соотношению, в которое входит открытый ключ. Однако, статистически неразличимые пары или наборы случайных значений могут быть выработаны нарушителем, используя только открытый ключ доказывающего.

Статистическая неразличимость (статистическая равнозначность) наборов случайных значений, формируемых нарушителем, от наборов, формируемых владельцем открытого ключа с использованием своего личного секретного ключа, понимается в том смысле, что эти наборы с одинаковым успехом могут использоваться в попытках вычисления по ним секретного ключа при знании открытого ключа.

Пусть, например, используется проверочное соотношение вида

$$yR = \alpha^w \bmod p, \qquad (1)$$

где p – достаточно большое простое число; α – число простого порядка r по модулю p; y – открытый ключ доказывающего, вычисляемый по личному секретному ключу x ($x < q$) и формуле $y = \alpha^x \bmod p$; R – значение фиксатора (разового открытого ключа), вырабатываемого по случайному равновероятному секретному значению k ($k < q$) и формуле $R = \alpha^k \bmod p$; w – ответ доказывающего, направляемый по открытому каналу связи.

Множество значений $\{\alpha^i \bmod p\}$, где $i = 1, 2, \ldots, q$, охватывает все возможные значения фиксатора R. При равновероятном случайном выборе значения k,

удовлетворяющего условию $k < q$, случайное значение R является равновероятным, т.е. принимает с одинаковой вероятностью любое значение из множества $\{\alpha^i \bmod p\}$. Нетрудно видеть, что ответ w вычисляется доказывающим по формуле $w = x + k \bmod q$, поэтому при равновероятном выборе k значение w является равновероятным, т.е. всевозможные значения w имеют одну и ту же вероятность, равную q^{-1}. Всевозможные пары значений (w, R), генерируемых доказывающим и удовлетворяющих соотношению (1), имеют одну и ту же вероятность, равную q^{-1}.

Такие же равновероятные пары (w, R) может сгенерировать и нарушитель. Для этого он генерирует случайное равновероятное значение $w < q$ и вычисляет значение $R = y^{-1}\alpha^w \bmod p$. При таком способе генерации пар (w, R), удовлетворяющих соотношению (1), всевозможные значения этих пар имеют одну и ту же вероятность q^{-1}. Таким образом, если наличие равновероятных случайных пар (w, R) из области их возможных значений как-то упрощают задачу дискретного логарифмирования по простому модулю, то нарушитель может самостоятельно сгенерировать равновероятные случайные пары (w, R) в нужном ему количестве без того, чтобы тратить время на ожидание выполнения протокола с нулевым разглашением.

3.2 Многораундовые протоколы

Многораундовые протоколы с нулевым разглашением включают многократное повторения трех типовых шагов:

1) генерация доказывающим разового случайного секретного ключа и вычисление по нему разового открытого ключа, который передается (объявляется) проверяющему и часто называется фиксатором;

2) генерация проверяющим нулевого ($r = 0$) или единичного ($r = 1$) запроса с вероятностью 0,5 и направление бита запроса r доказывающему;

3) вычисление доказывающим ответа w и направление w проверяющему.

После каждого такого трехшагового раунда проверяющий подставляет полученный им ответ в некоторое проверочное соотношения, в которое входят значения фиксатора, открытого ключа и запроса r. Если это соотношение выполняется то, проверяющий считает, что текущий ответ правильный.

Рассмотрим некоторые из данных протоколов.

Протокол Фиата–Шамира.

Протокол Фиата–Шамира основан на сложности извлечения квадратного корня по составному модулю, включающему не менее двух больших простых множителей, при условии, что разложение неизвестно. Доказывающий выбирает два больших простых числа p и q и вычисляет модуль $n = pq$. Затем выбирает в качестве своего личного секретного ключа случайное число s, такое, что $1 \le s \le n - 1$, и вычисляет значение $y = s^2 \bmod n$. (В дальнейшем он будет доказывать проверяющему то, что он знает квадратный корень из y.) Значение y, которое объявляется всем участникам протокола, играет роль открытого ключа в смысле его использования для проверки того, что доказывающий знает s.

Протокол состоит из z-кратного повторения раунда, включающего следующие три шага.

1. Доказывающий выбирает случайное число k, такое, что $1 \le k \le n - 1$, вычисляет значение $u = k^2 \bmod n$, называемое фиксатором, и посылает его проверяющему. (Число k играет роль разового секретного ключа и обеспечивает защиту личного секретного ключа от разглашения при направлении ответа, зависящего от s.)

2. Проверяющий отправляет доказывающему равновероятный случайный бит r ($r = 1$ или $r = 0$).

3. Доказывающий вычисляет значение $w = ks^r \bmod n$ и направляет его проверяющему. (Если $r = 1$, то $w = ks \bmod n$. Если $r = 0$, то $w = k$. Видно, что по этим двум результатам легко вычисляется секрет s, поэтому значения k должны уничтожаться после каждого раунда или после выполнения всего протокола.)

Проверяющий считает ответ верным, если выполняется соотношение

$$w^2 = uy^r \bmod n. \qquad (2)$$

Если $r = 1$, то должно выполняться $w^2 = uy \bmod n$. Если $r = 0$, то получаем $w^2 \bmod n = u$. В ходе осуществления протокола выполняется z шагов. Вероятность того, что нарушитель (который не знает секрета s) при выполнении одного раунда может дать положительный ответ, равна 2^{-1}, следовательно, вероятность того, что нарушитель может быть принят за пользователя, знающего секрет s, составляет 2^{-z}. Выбирая в протоколе достаточно большое число раундов проверки, можно сделать сколь угодно низкой вероятность обмана.

Рассмотрим два возможных варианта действий нарушителя в одном раунде.

В первом варианте он выбирает произвольное число k и передает проверяющему значение $u = k^2 \bmod n$. Если он получит от проверяющего запрос $r = 0$, то направит правильный ответ $w = k$. Однако правильно ответить на запрос $r = 1$ нарушитель не имеет возможности.

Во втором варианте нарушитель выбирает произвольное число k и направляет проверяющему число $u' = k^2/y$. Если он получит от проверяющего запрос $r = 1$, то направит ответ $w' = k$, который будет принят проверяющим за правильный, поскольку

$$u'y = (k^2/y)\,y = k^2 = w'^2 \bmod n.$$

Однако, на запрос $r = 0$ нарушитель правильно ответить не сможет.

Таким образом, нарушитель в лучшем случае может правильно ответить только на один вопрос, и в одном раунде с вероятностью 1/2 попытка обмана обнаруживается.

В протоколе нет утечки информации о ключе. Действительно, по запросу проверяющего $r = 0$ доказывающий направляет ему случайное число k, но проверяющий самостоятельно мог бы сгенерировать случайное число k', возвести его в квадрат, получив значение $u' = k'^2 \bmod n$. Самостоятельно полученная пара случайных чисел (u', k') ничем не хуже, чем пара случайных чисел (u, k), полученных от доказывающего. По запросу проверяющего $r = 1$ доказывающий направляет ему случайное число w, но проверяющий самостоятельно мог бы сгенерировать случайное число w' и вычислить случайное значение $u' = w'^2/y \bmod n$. Нетрудно видеть, что, если проверяющий сможет вычислить квадратный корень из случайного u', то потом он легко вычислит секретный ключ. Идентичную возможность имеет проверяющий при получении пары случайных чисел (u, w), связанных соотношением $w^2 = uy \bmod n$, т. е. если проверяющий сможет вычислить квадратный корень из случайного числа u, то затем он легко вычислит секретное значение s. Таким образом, данные, полученные в процессе выполнения описанного протокола проверяющим от доказывающего, не дают никаких новых возможностей проверяющему для вычисления секретного ключа. В этом смысле следует понимать то, что утечки информации о секрете, в ходе протокола, не происходит.

Протокол на основе трудности дискретного логарифмирования.

В данном протоколе открытый ключ доказывающего вычисляется по его личному секретному ключу x с использованием формулы $y = \alpha^x \bmod p$, где p – достаточно большое простое число; α - число достаточно большого простого порядка q по модулю p. Протокол с нулевым разглашением может включать z-кратное выполнения раунда, состоящего в выполнении следующих трех шагов:

1. Доказывающий выбирает текущий разовый секрет k ($k < q$), вычисляет значение фиксатора $R = \alpha^k \bmod p$ (который играет роль разового открытого ключа) передает число R проверяющему.

2. Проверяющий отправляет доказывающему равновероятный случайный бит r ($r = 1$ или $r = 0$) в качестве своего запроса.

3. Доказывающий вычисляет ответ на запрос в виде числа w по формуле $w = k + rx \bmod p$ и направляет его проверяющему.

Проверяющий проверяет выполнимость следующего проверочного соотношения

$$Ry^r \equiv \alpha^w \bmod p. \qquad (3)$$

Рассмотрим два возможных варианта действий нарушителя. Он может выбрать произвольное число k и передать проверяющему значение фиксатора $R = \alpha^k \bmod p$. При получении от проверяющего запроса $r = 0$ он направит правильный ответ $w = k$. Однако, правильного ответа на запрос $r = 1$ у нарушителя нет. В другом варианте нарушитель выбирает произвольное значение w и направляет проверяющему в качестве значения фиксатора число $R \equiv y^{-1}\alpha^w \bmod p$. Если он получит от проверяющего запрос $r = 1$, то направит ответ w, который будет принят проверяющим за правильный, поскольку будет выполняться проверочное уравнение (3). Однако, на запрос $r = 0$ нарушитель правильно ответить не сможет.

В каждом из двух вариантов его действий, нарушитель имеет возможность дать правильный ответ с вероятностью 1/2. Вероятность того, что нарушитель даст правильный ответ во всех z раундах равна 2^{-z}. Поскольку нарушитель не может навязывать многократное выполнение процедуры аутентификации абонентам, то значения $z = 24$; $z = 32$ и $z = 40$ могут быть вполне достаточными для многих практических применений.

3.3. Трехшаговые протоколы

Достоинством многораундового протокола Фиата–Шамира является его сравнительно низкая вычислительная сложность − каждая из сторон участвующих в протоколе выполняет не более $2z$ модульных умножений, где z – заданное число раундов. Однако, существенным недостатком всех многораундовых протоколов является необходимость выполнения очень большого числа чередующихся пересылок сообщений от доказывающего к проверяющему и обратно. Этот недостаток можно устранить, используя механизм объединения всех случайных однобитовых запросов проверяющего в единую случайную битовую строку, которая направляется доказывающему целиком, и свертки всех ответов в единое значение, которое направляется от доказывающего к проверяющему. То есть, доказывающий вычисляет z различных фиксаторов и отправляет их, в определенном порядке, проверяющему. Доказывающий получает от проверяющего запрос в виде равновероятной случайной z-битовой цепочки, для z соответствующих друг другу пар значений фиксатора и бита запроса вычисляет z ответов и высылает их проверяющему в соответствующем порядке. Таким образом, любой многораундовый

протокол может быть данным способом преобразован в трехшаговый протокол, сохраняя его исходную вычислительную сложность.

Некоторые многораундовые протоколы могут быть преобразованы в трехшаговые более практичным способом. Последний способ реализуется за счет использования открытого ключа, представляющего собой z упорядоченных значений, вычисляемых как независимые открытые ключи исходного многораундового протокола.

Трехшаговый вариант протокола Фиата–Шамира.

Личный секретный ключ каждого пользователя представляет собой два больших сильных простых числа p и q и z последовательных значений $(s_1, s_2, \ldots, s_z)$. Открытый ключ вычисляется в виде z упорядоченных значений $(y_1, y_1, \ldots, y_z)$ по формуле $y_i = s_i^2 \bmod n$, $i = 1, 2, \ldots, z$. Вычислительная трудность извлечения квадратного корня по составному модулю n имеет один порядок с вычислительной трудностью задачи разложения модуля n на простые множители, поэтому можно говорить, что описываемый далее протокол основан на вычислительной трудности задачи факторизации. Процедура проверки подлинности владельца открытого ключа, играющего роль доказывающего, выполняется за следующие три шага:

1. Доказывающий выбирает случайное число k, такое, что $1 \le k \le n - 1$, вычисляет значение $u = k^2 \bmod n$ и посылает его проверяющему.

2. Проверяющий отправляет доказывающему случайную равновероятную z-битовую строку $R = (r_1, r_2, \ldots, r_z)$ в качестве своего запроса, в которой каждый бит r_i с вероятностью 0,5 равен 1 и с вероятностью 0,5 равен нулю.

3. Доказывающий вычисляет свой ответ в виде значения $w = ks_1^{r_1} s_2^{r_2} \ldots s_z^{r_z} \bmod n$ и направляет его проверяющему.

Проверяющий считает ответ положительным, если выполняется соотношение

$$w^2 = uy_1^{r_1} y_2^{r_2} \ldots y_z^{r_z} \bmod n. \qquad (4)$$

Легко показать, что вероятность обмана проверяющего в этом протоколе составляет 2^{-z}.

Трехшаговый протокол на основе сложности дискретного логарифмирования.

Аутентификация пользователей выполняется по открытому ключу, представляющему собой набор из z значений $y_i = \alpha^{x_i} \bmod p$, где $i = 1,2,\ldots, z$; α - число достаточно большого простого порядка q по модулю p. Секретным ключом пользователя является набор значений x_i, $i = 1, 2,\ldots, z$. Протокол состоит из выполнения следующих трех шагов:

1. Доказывающий выбирает текущий разовый секрет k, вычисляет значение $R = \alpha^k \bmod p$ (которое играет роль разового открытого ключа) передает число R проверяющему.

2. Проверяющий генерирует свой случайный запрос в виде битовой последовательности $E = (e_1, e_2, \ldots, e_z)$ и направляет ее доказывающему.

3. Доказывающий вычисляет значение ответа на запрос в виде числа $w = k + \sum_{i=1}^{z} x_i e_i \bmod q$ и направляет его проверяющему.

Проверяющий считает полученный им ответ правильным, а доказывающего подлинным, если выполняется следующее проверочное соотношение

$$\alpha^{w} = R\prod_{i=1}^{z} y_i^{e_i} \bmod p\,. \qquad (5)$$

В правой части (5) присутствует произведение некоторого набора элементов открытого ключа, а именно, тех элементов, которые имеют номера, совпадающие с номерами единичных битов запроса.

Вероятность обмана проверяющего со стороны нарушителя, пытающегося выдать себя за подлинного владельца открытого ключа, равна 2^{-z}. Потенциальная атака со стороны нарушителя состоит в следующем: он выбирает случайное число w и случайное z-битовое значение $E' = (e'_1, e'_2, ..., e'_z)$, по которым вычисляет значение $R = \alpha^{w}\prod_{i=1}^{z} y_i^{-e_i} \bmod p$. Затем, выдавая себя за подлинного доказывающего, направляет значение фиксатора проверяющему. Последний генерирует случайный z-битовый запрос, который с вероятностью 2^{-z} совпадает со значение E'. Если такое совпадение имеет место, то он направляет заранее выбранный им ответ, при котором (5) выполняется, т.е. нарушитель будет принят за подлинного доказывающего.

Нетрудно показать статистическую равнозначность пар значений (R, w), формируемых доказывающим в ходе протокола с нулевым разглашением при случайном равновероятном значении запроса E и пар значений (R, w), формируемых нарушителем при равновероятном выборе значений w и E'. Действительно, всевозможные тройки значений (R, E, w) имеют место с одинаковой вероятностью, равной $2^{-z}q^{-1}$, в ходе выполнения протокола с участием подлинного доказывающего. Данные значения нарушитель может получить статистически равнозначными значениям (R, E', w). Также являются равновероятным, если нарушитель генерирует случайные значения E' и w. То есть существует способ генерации статистически равнозначных троек (R, E', w) без знания секретного ключа. Если такие тройки можно использовать для решения задачи дискретного логарифмирования и вычисления секретного ключа по данному открытому ключу, то это можно сделать, не дожидаясь, когда реальный протокол будет выполнен нужное число раз. Нарушитель может просто сгенерировать статистически равнозначные данные в нужном ему количестве и с одинаковым успехом приступить к решению задачи дискретного логарифмирования.

Раздел 4. Двухшаговые протоколы с нулевым разглашением секрета

4.1. Протоколы на основе алгоритмов открытого шифрования

Построение протоколов с нулевым разглашением можно реализовать, используя известные алгоритмы открытого шифрования. Для этого будет использоваться секретный ключ. Для того, кто выполнил процедуру шифрования правильно, восстановленное сообщение будет нести только информацию о том, что тот, кто восстановил зашифрованное сообщение, знает секретный ключ, связанный с открытым ключом. Однако, потенциальный нарушитель может выбрать произвольное значение и объявить его криптограммой, полученной в результате шифрования сообщения *M*, и попросить владельца открытого ключа выполнить процедуру расшифрования криптограммы. Если последний это сделает и раскроет восстановленное сообщение, то уже потенциально может иметь место утечка информации о секретном ключе. Говорить, что сообщение не несет в себе информацию о секретном ключе нельзя, так как была раскрыта информация. Поэтому, при построении протоколов с нулевым разглашением секрета, нужен некоторый механизм, который позволяет владельцу открытого ключа (доказывающему) убедиться до раскрытия восстановленного сообщения в том, что последнее уже известно (проверяющему).

В качестве такого механизма могут использоваться алгоритмы хэширования (хэш-функции) или заранее специфицированные метки, встраиваемые в исходные сообщения. Первый механизм используется в протоколах с нулевым разглашением, описанных в стандарте [3]. В протоколах такого типа предполагается использование некоторого алгоритма открытого шифрования *Publ_Encr*. Проверяющий может направить доказывающему некоторое случайное сообщение *M*, предварительно зашифровав его по открытому ключу доказывающего, т.е. направив доказывающему криптограмму $C = Publ\_Encr(M, P)$, где *P* – открытый ключ доказывающего. Прочитать это сообщение может только тот, кто знает секрет, связанный с открытым ключом *P*, т.е. только подлинный доказывающий. Получив значение *C* в качестве случайного запроса, доказывающий должен расшифровать криптограмму и направить проверяющему в качестве своего ответа исходное сообщение *M*, т.е. проверяющий получает значение которое уже знает. Поскольку проверяющий не получил никакого нового сообщения, то и утечки информации о личном секретном ключе доказывающего не происходит. При этом проверяющий получает информацию о том, что доказывающий является подлинным.

Чтобы проверяющий не имел возможности выбрать случайный запрос, а вынужден был его формировать путем выполнения процедуры шифрования. В стандарте [3] регламентируется формирование запроса в виде пары значений (*C, H*), где *C* – шифртекст, полученный путем шифрования некоторого сообщения *M* по открытому ключу доказывающего и *H* – значение хэш-функции, вычисленное от сообщения *M* с использованием некоторой специфицированной хэш-функции F_H: $H = F_H(M)$. Получая запрос (*C, H*), доказывающий имеет возможность убедиться в том, что восстановленное им из шифртекста *C* сообщение *M* известно проверяющему. Для этого достаточно вычислить значение хэш-функции от восстановленного сообщения и сравнить его со значением второго элемента запроса.

В соответствии с [3] двухшаговый протокол с нулевым разглашением секрета (с нулевыми знаниями) включает следующие шаги:

1. Проверяющий выбирает произвольное сообщение *M* и, используя специфицированный алгоритм открытого шифрования *Publ_Encr* и открытый ключ *P* доказывающего, зашифровывает сообщение *M* в шифртекст $C = Publ\_Encr(M, P)$. Затем, используя специфицированную хэш-функцию F_H, вычисляет значение хэш-функции от *M*:

$H = F_H(M)$. После этого он отправляет доказывающему пару значений (C, H) в качестве своего запроса.

2. Доказывающий расшифровывает криптограмму C, используя свой личный секретный ключ, в результате чего получает восстановленное сообщение M'. Затем он вычисляет значение хэш-функции от M': $H' = F_H(M')$, сравнивает значения H' и H и, если $H' = H$, то отправляет проверяющему значение M' в качестве своего ответа.

Поучив ответ M', проверяющий сравнивает M' и M. Если $M' = M$, то он делает вывод о подлинности доказывающего.

В работе [4] предложено построения двухшаговых протоколов с использованием механизма меток μ, включаемых в шифруемое сообщение. Наличие таких меток в восстановленном сообщение M' используется как свидетельство о том, что M' было известно проверяющему в момент формирования запроса ша первом шаге протокола. Размер меток $|\mu|$ выбирается равным от 80 до 512 бит, благодаря чему вероятность $\Pr(\mu)$ того, что расшифрование некорректно сформированного запроса даст текст, в котором будет присутствовать метка μ является пренебрежимо малой: $\Pr(\mu) = 2^{-|\mu|}$. Использование меток устраняет необходимость дополнительного использования хэш-функций и выполнения вычисления хэш-кода как проверяющим, так и доказывающим.

4.2. Протокол на основе использования криптосхемы RSA

Рассмотрим случай построения протокола с нулевым разглашением с использованием алгоритма открытого шифрования RSA [19], в котором открытый ключ формируется в виде пары чисел (n, e). Первое число представляет собой произведение двух сильных простых чисел r и q, генерируемых по случайному закону, а второе число представляет собой 32-битовое значение, которое выбирается таким, чтобы оно было взаимно простым с числом $L(n) = \text{НОК}[r - 1, q - 1]$, где НОК – наименьшее общее кратное чисел $r - 1$ и $q - 1$; $L(n)$ – значение обобщенной функции Эйлера от числа n. Секретный ключ вычисляется по формуле

$$d \equiv e^{-1} \bmod L(n).$$

После вычислении секретные значения r и q уничтожаются. Процедура открытого шифрования сообщения $M < n$ описывается формулой

$$C = M^{e} \bmod n.$$

Процедура расшифрования криптограммы C описывается формулой

$$M = C^{d} \bmod n.$$

Корректность процедуры расшифрования легко доказывается с использованием обобщенной теоремы Эйлера, согласно которой для любого числа M, взаимно простого с n имеет место соотношение

$$M^{L(n)} \equiv 1 \bmod n.$$

Рассмотрим двухпроходный протокол с нулевым разглашением на основе данного алгоритма открытого шифрования при использовании 128-битовой метки $\mu = n \bmod 2^{128}$. В качестве метки берутся 128 младших битов открытого ключа доказывающего. Протокол описывается следующими шагами:

1. Проверяющий генерирует случайное сообщение M размером $|M|$, удовлетворяющим условию $|n|/2 < |M| < |n|-|\mu|$. Затем он зашифровывает сообщение M с присоединенной к нему меткой μ, т.е. зашифровывает битовую строку $M||\mu$, по открытому ключу доказывающего (n, e), т.е. по формуле $C = (M||\mu)^{e} \bmod n$ и направляет

доказывающему значение C в качестве своего запроса, на который он ожидает ответ доказывающего.

2. Доказывающий расшифровывает криптограмму C по своему личному секретному ключу d по формуле $M'\|\mu' = C^d \bmod n$, где μ' – битовая строка, заданная младшими 256 битами расшифрованного значения, и проверяет выполнимость равенства $\mu' = \mu$. Если $\mu' = \mu$, то полученное значение M' доказывающий направляет проверяющему в качестве своего ответа на полученный запрос. В противном случае доказывающий отправляет ответ «Некорректный запрос».

Если проверяющий получил в качестве ответа правильное значение $M' = M$, т.е. то значение, которое он сгенерировал на первом шаге протокола (т.е. до направления своего запроса доказывающему), то делается вывод о подлинности доказывающего.

В описанном протоколе важным является использование заранее специфицированной метки μ, благодаря чему обеспечивается возможность доказывающему удостовериться в том, что отправляемое им значение ответа M' уже известно проверяющему. Тот факт, что оно уже известно проверяющему, означает, что проверяющий не пытается получить подпись к некоторому сообщению, используя механизм слепой подписи [5,6], и не пытается выполнить атаку при адаптивно выбираемом шифртексте (adaptive chosen cipher text attack), описанную в работе [7] или какую-то другую атаку.

4.3. Использование алгоритма открытого шифрования Эль-Гамаля

Рассмотрим реализацию прокола с нулевым разглашением секрета, основанном на использовании алгоритма открытого шифрования Эль-Гамаля [2]. В алгоритме используются открытый ключ вида $y = \alpha^x \bmod p$, где p – большое простое число (размером 1024 бит и более), такое, что разложение числа $p-1$ содержит простой делитель разрядностью не менее 160 бит; α – примитивный элемент по модулю p. Этот способ фактически представляет собой гибридную криптосистему, в которой секретные ключи распределяются в соответствии с протоколом Диффи–Хеллмана [8], а шифрование сообщения выполняется путем модульного умножения сообщения на секретный ключ. Шифрование сообщения T, отправляемого владельцу открытого ключа y, осуществляется с помощью следующего алгоритма:

1. Сгенерировать случайное число k, которое по своей сути является разовым секретным ключом отправителя сообщения.

2. Вычислить число $R = \alpha^k \bmod p$ – разовый открытый ключ отправителя.

3. Используя открытый ключ получателя y, вычислить разовый общий секретный ключ $Q = y^k \bmod p$.

4. Зашифровать сообщение M путем умножения сообщения на разовый секретный ключ: $C = QM \bmod p$.

5. Отправить получателю криптограмму в виде пары чисел (R, C).

Процедуру открытого шифрования по открытому ключу y обозначим как *Gamal_Encr*(M, y). Получатель криптограммы (R, C), используя свой личный секретный ключ x, выполняет процедуру расшифрования *Gamal_Decr*(R, C, x), которая описывается следующими шагами:

1. Вычислить разовый общий секретный ключ $Q = R^x \bmod p$.

2. Используя расширенный алгоритм Евклида, вычислить значение Q^{-1}, обратное значению Q по модулю p.

3. Расшифровать сообщение M путем умножения значения C на целое число Q^{-1}: $M = CQ^{-1} \bmod p$.

При шифровании сообщений в алгоритме Эль-Гамаля используются случайные значения, т.е. он реализует процедуру вероятностного шифрования, при которой одному и тому же сообщению соответствует множество различных криптограмм, все из которых, при расшифровании, дают одно и то же значение M.

По аналогии с протоколом, предложенным в [4], построим протокол аутентификации с нулевым разглашением на основе алгоритма Эль-Гамаля при использовании 256-битовой метки μ, в качестве которой берутся младшие 160 бит открытого ключа доказывающего y (т.е. специфицируется значение $\mu = y \bmod 2^{160}$), который описывается следующим образом:

1. Проверяющий генерирует случайное сообщение M размером $|M|$, удовлетворяющим условию $|M| < |p|-|2\mu|$. Затем он зашифровывает сообщение M с присоединенной к нему меткой μ, т.е. зашифровывает битовую строку $M||\mu$, по открытому ключу доказывающего y в соответствии с формулой $(R, C) = Gamal\_Encr(M||\mu, y)$. Затем он направляет доказывающему пару (R, C) в качестве своего запроса, на который доказывающий должен дать ответ.

2. Доказывающий расшифровывает криптограмму (R, C) по своему личному секретному ключу x: $M'||\mu' = Gamal\_Decr(R, C, x)$, где μ' – битовая строка, заданная младшими 160 битами расшифрованного значения, и проверяет выполнимость равенства $\mu' = \mu$. Если $\mu' = \mu$, то полученное значение M' доказывающий направляет проверяющему в качестве своего ответа на полученный запрос. В противном случае доказывающий отправляет ответ «Некорректный запрос».

Если проверяющий не имеет своей целью получение информации о секретном ключе доказывающего, то он корректно выполняет первый шаг протокола с использованием специфицированной метки μ, которую он присоединит к выбираемому им сообщению M, а потом выполнит процедуру зашифрования. В этом случае при расшифровании шифртекста доказывающий восстановит исходное сообщение, увидит в нем специфицированную метку, отбросит ее и получит значение $M' = M$. Проверяющий получает от доказывающего известное ему значение и делает вывод о подлинности доказывающего. Вероятность выполнения соотношения $\mu' = \mu$, для произвольно выбранного запроса, пренебрежимо мала и равна значению 2^{-160}.

4.4. Протокол на основе криптосхемы Рабина

В алгоритме открытого шифрования Рабина [9] используются вычисления по модулю вида $n = pq$, который используется в качестве открытого ключа. Сильные простые числа p и q составляют личный секретный ключ и удовлетворяют условиям: $p \equiv 3 \bmod 4$ и $q \equiv 3 \bmod 4$. Последние два условия обеспечивает снижение сложности процедуры расшифрования криптограммы по сравнению со случаем выбора значений p и q, которые не удовлетворяют указанным двум условиям.

Шифрование сообщения $M < n$ выполняется возведением числа M в квадрат по модулю n:

$$C = M^2 \bmod n.$$

Процедура расшифрования состоит в извлечении квадратного корня из криптограммы C по модулю n. Предварительно вычисляют корни из C по модулям p и q:

$$m_{p_1} = C^{\frac{p+1}{4}} \bmod p, \quad m_{p_2} = p - m_{p_1} = p - C^{\frac{p+1}{4}} \bmod p;$$

$$m_{q_1} = C^{\frac{q+1}{4}} \bmod q, \quad m_{q_2} = q - m_{q_1} = q - C^{\frac{q+1}{4}} \bmod q.$$

Из этих четырех значений вычисляются четыре возможных корня из C по модулю n:

$$M_1 = \left(m_{p_1} a + m_{q_1} b \right) \bmod n,$$

$$M_2 = \left(m_{p_1} a + m_{q_2} b \right) \bmod n,$$

$$M_3 = \left(m_{p_2} a + m_{q_1} b \right) \bmod n,$$

$$M_4 = \left(m_{p_2} a + m_{q_2} b \right) \bmod n,$$

где $a = q(q^{-1} \bmod p)$ и $b = p(p^{-1} \bmod q)$.

Двухпроходный протокол с нулевым разглашением на основе алгоритма открытого шифрования Рабина использует 128-битовую метку μ, в качестве которой берутся младшие 128 бит открытого ключа доказывающего, т.е. $\mu = n \bmod 2^{128}$, и имеет следующий вид:

1. Проверяющий генерирует произвольное сообщение M размером $|M|$, удовлетворяющим условию $|n|/3 < |M| < |n|-|2\mu|$, и зашифровывает сообщение M с присоединенной к нему меткой μ, т.е. значение $M||\mu$, по формуле $C = (M||\mu)^2 \bmod n$ и направляет доказывающему значение C в качестве своего запроса.

2. Доказывающий расшифровывает криптограмму C по своему секретному ключу (p, q) путем вычисления четырех значений квадратного корня из C: M_1, M_2, M_3 и M_4. Каждое из последних значений он представляет в виде $M_i' || \mu_i'$, где μ_i' – битовая строка, заданная младшими 128 битами i-го корня (i = 1,2,3,4). Затем он проверяет, выполняется ли равенство $\mu_i' = \mu$ для одного из четырех значений i. Если да, то соответствующее значение M_i' доказывающий направляет проверяющему в качестве своего ответа на полученный запрос. Если $\mu_i' \neq \mu$ для i = 1,2,3,4, то доказывающий отправляет ответ «Некорректный запрос».

Использование заранее специфицированной метки позволяет доказывающему идентифицировать исходное сообщение, благодаря чему он направляет проверяющему именно то значение, которое известно проверяющему. Это обеспечивает нулевую утечку информации о секретном ключе при передаче доказывающим своего ответа проверяющему.

Раздел 5. Протокол на основе схемы открытого согласования ключа

5.1. Реализация с использованием криптосхемы Диффи–Хеллмана

Стойкие и удобные в использовании протоколы строгой аутентификации удаленных пользователей могут быть построены на основе использования схем открытого согласования ключей. Рассмотрим вариант реализации с использованием схемы Диффи–Хеллмана, в которой системными параметрами являются большое простое число p и примитивный элемент α по модулю p. В этой схеме каждый пользователь выбирает случайный секретный ключ x и вычисляет открытый ключ y по формуле $y = \alpha^x \bmod p$.

Открытый ключ делается общеизвестным и любой желающий имеет принципиальную возможность однозначно вычислить значение секретного ключа x, хотя эта возможность практически нереализуема, если число p имеет размер, например, не менее 3072 бит, а число $p - 1$ содержит простой делитель q размером не менее 256 бит. Рассмотрим протокол аутентификации удаленных пользователей с нулевым разглашением, который использует значение α, имеющее по модулю p порядок, равный q, и включает следующие два шага:

1. Проверяющий генерирует случайное число k и вычисляет значение своего разового открытого ключа $U = \alpha^k \bmod p$ и значение $Z = y^k \bmod p$, где y – открытый ключ доказывающего (т.е. пользователя, подлинность которого проверяется), после чего передает доказывающему значение U в качестве своего запроса, на который он ожидает ответ доказывающего.

2. Доказывающий проверяет выполнимость соотношения $U^q \bmod p = 1$ (проверка того, что q является порядком числа U по модулю p). Если да, то он вычисляет значение $Z = U^x \bmod p$ и направляет Z проверяющему в качестве своего ответа на полученный запрос. Если указанное соотношение не выполняется, то доказывающий направляет ответ «Некорректный запрос».

Если проверяющий получил правильное значение Z, т.е. то значение, которое он вычислил до направления своего запроса доказывающему, то им делается вывод о подлинности доказывающего. Проверка на втором шаге числа U является принципиальным моментом. Если эта проверка не делается, то становится возможна атака со стороны проверяющего, результатом которой будет вычисление части или всего секретного ключа доказывающего. Действительно, это можно сделать, посылая в качестве запроса значения U', порядок которых q' содержит только небольшие простые делители числа $p - 1$. Используя ответ $Z' = U'^x \bmod p$ и алгоритм больших и малых шагов [5,10], можно сравнительно легко вычислить значение $x' = x \bmod q'$. Получив несколько соотношений последнего вида и используя китайскую теорему об остатках, можно легко вычислить секретный ключ x.

В рассмотренном двухшаговом протоколе проверяющий может сформировать в качестве своего запроса, случайное или специально подобранное число U'', порядок которого равен q, и получить ответ доказывающего, равный $Z'' = U''^x \bmod p$, который заранее не был ему известен. Однако, в этом случае утечки информации о секретном ключе не происходит. Это можно показать следующим образом. Допустим, что проверяющий знает значение степени k'', такое, что имеет место $U'' = \alpha^{k''} \bmod p$. При нашем предположении проверяющий самостоятельно может вычислить значение Z''. То есть, при наличии у проверяющего дополнительной информации к той информации, которой он обладает, ответ доказывающего не передает проверяющему никакой

информации о секретном ключе. Последнее означает, что и в отсутствии дополнительной информации имеет место нулевая утечка информации о секретном ключе.

Чтобы не обременять себя усвоением такого доказательства нулевой утечки знаний о секретном ключе, можно реализовать протокол, основанный на схеме открытого согласования ключа, с использованием вычисления значения хэш-функции, вычисляемой от общего разового секретного ключа Z. В этом случае протокол использует некоторую специфицированную хэш-функцию F_H и включает следующие два шага:

1. Проверяющий генерирует случайное число k и вычисляет значение своего разового открытого ключа $U = \alpha^k \bmod p$ и значение $Z = y^k \bmod p$, где y – открытый ключ доказывающего (т.е. пользователя, подлинность которого проверяется). Затем он вычисляет значение хэш-функции от Z: $H = F_H(Z)$ и направляет доказывающему пару значений (U, H) в качестве своего запроса.

2. Доказывающий вычисляет значение $Z = U^x \bmod p$ и значение $H' = F_H(Z)$. Затем он сравнивает значения H' и H. Если $H' = H$, то он направляет Z проверяющему в качестве своего ответа на полученный запрос. Если $H' \neq H$, то доказывающий направляет ответ «Некорректный запрос».

Недостатком использования хэш-функции является то, что требуется использование еще одного алгоритма. Это обстоятельство может иметь существенное значение при аппаратной реализации. Однако операция вычисления значения хэш-функции имеет меньшую вычислительную сложность, чем дополнительная операция возведения значения U в большую степень q по модулю p, которая выполняется для проверки того, что значение запроса по модулю p имеет порядок, равный q.

Легко понять, что если первый вариант протокола, основанного на схеме открытого согласования ключа, реализовать с использованием конечной циклической группы простого порядка, то в нем можно отказаться от проверки того, что полученный запрос имеет порядок, равный q, поскольку все значения такой группы имеют порядок, равный значению q. Возможный вариант реализации двухшаговых протоколов с нулевым разглашением с использованием групп простого порядка рассмотрен в следующем подразделе.

5.2. Реализация с использованием простых чисел Мерсенна

В качестве циклических групп простого порядка можно использовать мультипликативные группы конечных полей $GF(2^s)$ двоичных многочленов для некоторых значений степени s. Порядок мультипликативной группы таких полей равен $q = 2^s - 1$. Значение $2^s - 1$ является простым, если значение s равно степени Мерсенна. (Степень Мерсенна – это такое простое число s, при котором значение $2^s - 1$ тоже является простым числом.)

Для следующих значений степени $s \geq 1024$ имеем простые значения q: 1279, 2203, 2281, 3217, 4253, 4423, 9689, 9941, 11213, 19937, 21701, 23209, … [11,12]. Найдено еще много других значении $s \geq 44497$ [12], задающих простые числа Мерсена, однако они в настоящее время представляют меньший интерес для использования в протоколах с нулевым разглашением секрета, поскольку это приводит к существенному увеличению вычислительной сложности протокола. Операцией умножения в полях $GF(2^s)$ является операция умножения многочленов по модулю неприводимого многочлена степени s. Для уменьшения вычислительной сложности умножения в поле $GF(2^s)$ следует выбрать неприводимый многочлен степени s, имеющий малый вес (малое число ненулевых коэффициентов) и выполнять модульное умножение многочленов. Таблица 1 показывает примеры неприводимых двоичных трехчленов, степени которых равны степеням

Мерсенна. Данные многочлены представляют интерес для использования при реализации протоколов аутентификации с нулевым разглашением секрета на основе схем открытого согласования ключа.

Таблица 1: Неприводимые двоичные многочлены малого веса.

Степень Мерсенна	Двоичные многочлены
1279	$x^{1279} + x^{216} + 1;\ \ x^{1279} + x^{418} + 1$
2203	$x^{2203} + x^{14} + x^{6} + x^{5} + 1$
2281	$x^{2281} + x^{715} + 1;\ \ x^{2281} + x^{915} + 1;$
3217	$x^{3217} + x^{67} + 1;\ \ x^{3217} + x^{576} + 1;$
4423	$x^{4423} + x^{271} + 1;\ \ x^{4423} + x^{369} + 1;\ \ x^{4423} + x^{370} + 1;$
9689	$x^{9689} + x^{84} + 1;\ \ x^{9689} + x^{471} + 1;$
9941	$x^{9941} + x^{29} + x^{12} + x^{10} + 1$
11213	$x^{11213} + x^{8218} + x^{6181} + x^{2304} + 1$
19937	$x^{19937} + x^{881} + 1;\ \ x^{19937} + x^{7083} + 1$

Практический интерес для реализации протокола с нулевым разглашением представляют, например, следующие значения степени $s = 1279$ и $s = 2203$. Примерами неприводимых двоичных многочленов таких степеней являются трехчлены $\eta_1(x) = x^{1279} + x^{216} + 1$ и $\eta_2(x) = x^{1279} + x^{418} + 1$ [11], а также пятичлен $\eta_3(x) = x^{2203} + x^{14} + x^{6} + x^{5} + 1$. При выборе указанных трехчленов $\eta_1(x)$ и $\eta_2(x)$ в качестве модуля операция модульного умножения двоичных многочленов в поле $GF(2^{1279})$ может быть выполнена путем осуществления одной операции арифметического умножения многочленов, двух операций арифметического сдвига и шести операций сложения. Выполнение операции деления многочленов не требуется, что существенно снижает сложность вычислений в поле $GF(2^{1279})$ и повышает быстродействие протокола аутентификации. В поле $GF(2^{2203})$ операция умножения может быть задана по модулю неприводимого пятичлена $\eta_3(x)$. В этом случае умножение в поле $GF(2^{2203})$ может быть выполнено также без осуществления операции деления многочленов, так как оно реализуется путем осуществления одной операции арифметического умножения многочленов, шести операций арифметического сдвига и десяти операций сложения.

В двухшаговом протоколе аутентификации на основе трудности задачи дискретного логарифмирования в поле $GF(2^{1279})$ открытый ключ формируется по формуле $y(x) = (\alpha(x))^s \bmod \eta_1(x)$, где s – секретный ключ; $\alpha(x)$ – некоторый специфицированный многочлен, а протокол описывается следующими шагами:

1. Проверяющий генерирует случайное число k и вычисляет запрос в виде многочлена $u(x) = (\alpha(x))^k \bmod \eta_1(x)$ и многочлен $z(x) = (y(x))^k \bmod \eta_1(x)$, после чего пересылает доказывающему двоичный многочлен $u(x)$ в качестве своего запроса.

2. Доказывающий вычисляет двоичный многочлен $z'(x) = \left(u(x)\right)^s \bmod \eta_1(x)$ и направляет $z'(x)$ проверяющему в качестве ответа.

Если $z'(x) = z(x)$, то проверяющий делает вывод о подлинности доказывающего. В этом варианте протокола нет необходимости выполнять проверку корректности запроса, поскольку любая ненулевая битовая строка, интерпретируемая как элемент поля $GF(2^{1279})$ имеет порядок равный простому числу $2^{1279} - 1$, т.е. никакие запросы со стороны проверяющего не могут быть им использованы для получения хотя бы одного бита информации о ключе.

Практический интерес представляет реализация этого протокола с использованием идеальных эллиптических кривых (все точки которых, кроме бесконечно удаленной точки, имеют одно и то же значение порядка, равное простому числу) вместо конечных мультипликативных групп двоичных многочленов. Такая замена используемой конечной группы простого порядка обеспечивает существенное уменьшение вычислительной сложности протокола при заданном уровне стойкости.

Раздел 6. Протокол на основе последовательного возведения в квадрат

6.1. Протокол на основе последовательного возведения в квадрат

Рассмотрим двухпроходный протокол [13] с нулевым разглашением, основанный на трудности задачи факторизации, для которого доказательство нулевой утечки секрета является достаточно очевидным. В качестве открытого ключа доказывающего используется натуральное число n, равное произведению двух больших простых чисел r и q, составляющих его личный секрет. Идея доказательства отсутствия утечки информации о секретном ключе состоит в том, что доказывающий передает проверяющему значение, которое вычислено до того, как оно было вычислено доказывающим, поэтому никакой новой информации проверяющему не передается. Протокол включает следующие два шага:

1. Проверяющий генерирует случайное число $a < n$ и ρ-битовое число k, где $\rho = 24^{-40}$ (значение ρ выбирается с учетом времени задержки δ, вносимого каналом связи). Затем используя метод последовательного возведения в квадрат, вычисляет значение $T = a^{2^k} \bmod n$, после чего передает доказывающему пару значений a и k в качестве своего запроса, на который он ожидает ответ доказывающего;

2. Доказывающий выполняет две последовательные операции возведения в степень, в результате чего за короткое время вычисляет значения $K = 2^k \bmod L(n)$, где $L(n)$ – обобщенная функция Эйлера от числа n, и $T = a^K \bmod n$. Затем он сразу направляет проверяющему значение T в качестве своего ответа на полученный запрос.

Если проверяющий получил правильное значение T, т.е. то значение, которое он вычислил до направления запроса доказывающему, в течение временного интервала, длительность которого не превышает некоторое пороговое значение Δ, например $\Delta = 10\delta–1000\delta$ (где δ - время задержки передачи сообщения, вносимое каналом связи), то им делается вывод о подлинности доказывающего. Значение Δ выбирается исходя из того, что оно должно существенно превышать значение δ. Разрядность числа k выбирается такой, что процесс вычисления значения T без знания факторизации модуля требует у потенциального нарушителя выполнения вычислений в течение времени, длительность которого превышает значение Δ.

Без знания факторизации модуля вычисление значения $T = a^{2^k} \bmod n$ выполняется методом последовательного возведения в квадрат. При этом требуется выполнить k возведений в квадрат (умножений) по модулю n.

6.2. Особенности практического применения

Практическое применение протокола аутентификации, описанного в предыдущем подразделе, требует учета возможных вычислительных ресурсов у потенциального нарушителя. Если предполагается возможность применения нарушителем специализированных производительных ЭВМ (применение многопроцессорных ЭВМ не дает эффекта, так как процесс последовательного возведения в квадрат не может быть распараллелен), то проверяющему требуется выбирать более длинные числа k. При этом ему потребуется потратить больше времени на вычисление значения T. Выбираемая разрядность k определяется также и быстродействием канала связи, используемого в протоколе. Чем более быстродействующий канал, тем меньшее значение Δ может быть выбрано, т.е. тем меньшая разрядность числа k может быть использована. Это позволит уменьшить время вычислений, выполняемых проверяющим до направления запроса доказывающему.

Обычно, на практике, один пользователь связывается со многими другими пользователями, подлинность которых он желает проверить. Это означает, что он должен для каждого из проверяемых установить свое пороговое значение Δ или взять общее пороговое значение $\Delta_{общ}$, равное максимальному времени, требуемому для получения ответа, по всем проверяемым пользователям. В первом случае требуется индивидуальная настройка параметров протокола аутентификации, но достигается меньшее среднее время вычислений на первом шаге. Во втором случае устраняется необходимость индивидуальной настройки параметров, но увеличивается время вычислений на первом шаге.

При практическом использовании этого протокола может быть применен способ предвычислений, который состоит в том, что проверяющий заблаговременно вычисляет множество значений (a, k, T), которые впоследствии будут использоваться при осуществлении аутентификации удаленных пользователей.

Следует заметить, что использование технического параметра канала связи, связанного с быстродействием, вносит существенные ограничения на область применения данного протокола.

Раздел 7. Преобразование протоколов с нулевым разглашением в схемы цифровой подписи

7.1 Общий подход к выводу схем цифровой подписи и протоколов с нулевым разглашением

Протоколы с использованием фиксатора включают следующие три шага:

1. Генерация доказывающим значения фиксатора и направление его проверяющему.

2. Генерация проверяющим случайного запроса и отправка его доказывающему.

3. Вычисление доказывающим ответа на запрос и передача ответа проверяющему.

Правильность ответа проверяется по некоторому проверочному соотношению. В случае многораундовых протоколов эти три шага выполняются многократно. Любой протокол с нулевым разглашением, который можно отнести к этому типу, может быть преобразован в схему цифровой подписи. Идея такого преобразования состоит в использовании хэш-функции подписываемого документа в качестве одного случайного z-битового запроса (в трехшаговых протоколах) или битовой цепочки запросов, которую можно рассмотреть как z запросов проверяющего (в многораундовых протоколах). В соответствии с протоколом с нулевым разглашением до формирования запроса доказывающий формирует значение одного или z различных фиксаторов и после получения запроса доказывающий не может поменять значение фиксатора. Поэтому, рассматривая некоторый документ как источник случайного запроса, нужно использовать такой механизм формирования запроса, который требует использования значения фиксатора, т.е. запрос должен формироваться не только в зависимости от каждого бита документа, но и в зависимости от каждого бита фиксатора.

Это требование реализуется достаточно просто – использованием в качестве запроса значения хэш-функции от подписываемого документа с присоединенным к нему значением фиксатора в случае трехшаговых протоколов (или z различных фиксаторов в случае многораундовых протоколов). После того как значение хэш-функции вычислено, то замена фиксатора становится вычислительно невозможной, т.к. любое изменение в фиксаторе приводит к случайному изменению запроса (при этом теоретически возможно, когда значение запроса не изменится, однако вероятность этого пренебрежимо мала, если использовать хэш-функции с 160-битовыми или 256-битовыми значениями). Также невозможна модифицирование или замена подписываемого документа. Далее в трехшаговом протоколе с нулевым разглашением владелец открытого ключа (подписант) формирует ответ, используя свой личный секретный ключ. И если ответ удовлетворяет проверочному соотношению, то ответ признается правильным. Нетрудно понять, что правильность ответа по запросу, исходящему от документа, может проверить любой пользователь, если ему предоставить значение фиксатора, документ, ответ на запрос (само значение запроса не требуется, так как оно может быть вычислено тем, кто проверяет правильность ответа) и открытый ключ подписывающего. Пара значений фиксатора и ответа представляют собой цифровую подпись к документу, в зависимости от которого формировался ответ.

Аналогично, это можно сделать, и в случае использования многораундовых протоколов с нулевым разглашение, однако вместо одного фиксатора надо рассматривать z различных фиксаторов, вместо одного ответа – z различных ответов. То есть и в этом случае получаем преобразование указанного протокола в схему цифровой подписи, однако размер подписи будет в z раз превышать размер подписи в случае преобразования трехпроходного протокола.

Схемы ЭЦП, построенные с использованием преобразования протокола с нулевым разглашением, можно отнести к доказуемо стойким схемам цифровой подписи, поскольку предположение о нестойкости схемы цифровой подписи, использующей стойкую хэш-функцию, влечет за собой вывод о нестойкости протокола с нулевым разглашением, из которого была выведена схема цифровой подписи.

7.2 Схема цифровой подписи Фиата–Шамира

Эта схема цифровой подписи основана на сложности вычисления квадратных корней по модулю n, являющимся произведением двух больших простых чисел p и q [14], и ее можно представить как результат преобразования трехшагового протокола с нулевым разглашением Фиата – Шамира, описанного в разделе 3.3.

Возможны два варианта:

1) каждый абонент формирует свой индивидуальный модуль $n = pq$

2) все абоненты используют одно и то же значение составного модуля n, которое вырабатывается доверительным центром, причем исходные простые множители p и q уничтожаются (для формирования открытых и закрытых ключей не требуется знать разложение модуля).

Первый случай является предпочтительным, поскольку может обеспечить гарантии абоненту в том, что разложение модуля неизвестно потенциальному нарушителю.

Формирование секретного ключа осуществляется следующим образом. Абонент генерирует z случайных чисел $x_0, x_1, \ldots, x_i, \ldots, x_{z-1}$, таких что $(n \operatorname{div} 2^{64}) \le x_i \le n - 1$ и НОД$(x_i, n) = 1$ для $i = 0, 1, \ldots, z - 1$.

Формирование открытого ключа осуществляется путем вычисления значений, обратных квадратам чисел x_i по модулю n. Открытым ключом является набор чисел $y_0, y_1, \ldots, y_i, \ldots, y_{z-1}$, для которых имеем $y_i = x_i^{-2} \bmod n$.

Вычисление подписи к сообщению m включает следующие шаги.

1. Генерируется случайное число k, $k < n - 1$. В определенном смысле можно сказать, что k играет роль разового секретного ключа, обеспечивающего маскирование основного (долговременного) секретного ключа.

2. Вычисляется значение фиксатора $\mathrm{R} = \mathrm{k}^2 \bmod \mathrm{n}$ (по своей сути это разовый открытый ключ).

3. К сообщению M присоединяется число R и от полученного нового сообщения вычисляется z-битовое значение некоторой специфицированной хэш-функции F_H: $E = F_H(M||R)$, которое является первой частью подписи (случайный запрос, формируемы по значению подписываемого документа). Значение E представляется в виде последовательности битов e_i: $E = (e_0, e_1, \ldots, e_i, \ldots, e_{z-1})$. На данном шаге обеспечивается задание зависимости значения E от случайного числа k и от сообщения M.

4. Вычисляется вторая часть подписи (ответ на случайный запрос) в соответствии со следующей формулой:

$$S = k \prod_{i=0}^{z-1} x_i^{e_i} \bmod n .$$

На этом завершается формирование подписи (E, S) к сообщению M. Нетрудно видеть, что к фиксированному сообщению могут быть выработаны различные значения подписи, поскольку при ее формировании используется случайное число k.

Проверка подлинности подписи (проверка правильности ответа на случайный запрос) осуществляется следующим образом.

1. Вычисляется значение R':

$$R' = S^2 \prod_{i=0}^{z-1} y_i^{e_i} \bmod n .$$

2. К сообщению M присоединяется число R' и вычисляется значение хэш-функции $E' = F_H(M||R')$.

3. Сравниваются значения E и E'. Если $E = E'$, то подпись признается подлинной.

Заметим, что в качестве цифровой подписи можно было бы взять пару значений (R, S), а значение запроса E вычислять по значениям R и S. Однако, в таком варианте подпись получается в два раза длиннее.

Достоинством рассмотренной схемы ЭЦП является сравнительно малая сложность процедуры формирования и проверки подписи, что обеспечивает достаточно высокое быстродействие. Недостатком является большой размер открытого и закрытого ключей. Для 1024-битового модуля и 160 битовой хэш-функции размер каждого из ключей составит около 164 Кбит. Этот недостаток приводит к тому, что размер справочника открытых ключей может оказаться весьма объемным при достаточно большом числе абонентов. Кроме того, существенно усложняется аппаратная реализация схемы подписи.

Уменьшение размера открытого ключа в схеме Фиата – Шамира.

Для уменьшения размера открытого ключа в схеме подписи Фиата–Шамира можно применить достаточно простой метод – формирование z секретных значений $x_0, x_1, \ldots, x_i, \ldots, x_{z-1}$ по одному случайному значению x, которое представляет собой личный секретный ключ подписывающего, в соответствии с формулой

$$x_i = x^{2^i} \bmod n, \ i = 0,1,2,\ldots, z-1.$$

Открытым ключом является значение $y = x^{-2} \bmod n$, которое определяет z значений $y_0, y_1, \ldots, y_i, \ldots, y_{z-1}$, вычисляемых по формуле

$$y_i = x_i^2 \bmod n, \ i = 0,1,2,\ldots, z-1.$$

или по формуле

$$y_i = y^{2^i} \bmod n, \ i = 0,1,2,\ldots, z-1.$$

Имея в виду упорядоченные множества значений $(x_0, x_1, \ldots, x_i, \ldots, x_{z-1})$ и $(y_0, y_1, \ldots, y_i, \ldots, y_{z-1})$, вычисляемых по секретному ключу x и по открытому ключу y, соответственно, можно полностью сохранить в исходном виде все шаги процедуры генерации и проверки подлинности подписи схемы Фиата–Шамира.

При указанном способе вычисления значений x_i и y_i вычисление произведения значений x_i (также и y_i), выбираемых в зависимости от битов значения E, заменяется на одну операцию возведения секретного ключа x (открытого ключа y) в степень E по модулю n. Действительно, на шаге 4 процедуры вычисления подписи имеем

$$S \equiv k \prod_{i=0}^{z-1} x_i^{e_i} \equiv k \prod_{i=0}^{z-1} x^{2^i e_i} \equiv k x^{\sum_{i=0}^{z-1} 2^i e_i} \equiv k x^E \bmod n.$$

На шаге 1 процедуры проверки подлинности подписи имеем

$$R' \equiv S^2 \prod_{i=0}^{z-1} y_i^{e_i} \equiv S^2 \prod_{i=0}^{z-1} y^{2^i e_i} \equiv S^2 y^{\sum_{i=0}^{z-1} 2^i e_i} \equiv S^2 y^E \bmod n,$$

где битовая цепочка E рассматривается как двоичное число.

С учетом сделанных замечаний, получаем модифицированную версию схемы подписи Фиата–Шамира, которая описывается следующим образом.

Формирование секретного ключа осуществляется как генерация случайного числа x, для которого выполняется условие $(n \text{ div } 2^{64}) \le x \le n - 1$ и НОД$(x_i, n) = 1$.

Формирование открытого ключа осуществляется путем вычисления значения, обратного квадрату числа x по модулю n. Открытым ключом является число $y = x^{-2} \bmod n$.

Вычисление подписи к сообщению m включает следующие шаги.

1. Генерируется случайное число k, $k < n - 1$.

2. Вычисляется значение $R = k^2 \bmod n$.

3. К сообщению m присоединяется число R и от полученного нового сообщения вычисляется z-битовое значение некоторой специфицированной хэш-функции F_H: $E = F_H(M||R)$, которое является первой частью подписи. Значение E представляется в виде последовательности битов e_i: $E = (e_0, e_1, \ldots, e_i, \ldots, e_{z-1})$.

4. Вычисляется вторая часть подписи в соответствии со следующей формулой:

$$S = k \cdot x^E \bmod n.$$

Подписью к сообщению m является пара чисел (E, S). Нетрудно видеть, что к фиксированному сообщению могут быть выработаны различные значения подписи, поскольку при ее формировании используется случайное число k.

Проверка подлинности подписи осуществляется следующим образом.

1. Вычисляется значение R':

$$R' = S^2 y^E \bmod n.$$

2. К сообщению m присоединяется число R' и вычисляется значение хэш-функции $E' = F_H(M||R')$.

3. Сравниваются значения E и E'. Если $E = E'$, то подпись признается подлинной. Действительно, имеем:

$$R' = S^2 y^E \bmod n = k^2 (x^E)^2 (x^{-2})^E \bmod n = k^2 \bmod n = R.$$

7.3 Схема цифровой подписи на основе вычислительной трудности задачи дискретного логарифмирования

Рассмотрим преобразование трехшагового протокола с нулевым разглашением, основанного на вычислительной трудности задачи дискретного логарифмирования, из раздела 3.3, в схему цифровой подписи. В соответствии с исходным протоколом открытым ключом является упорядоченный набор $(y_0, y_1, \ldots, y_i, \ldots, y_{z-1})$ из z значений y_i, вычисляемых по формуле $y_i = \alpha^{x_i} \bmod p$, где $i = 1,2,..., z$; α - число достаточно большого простого порядка q по модулю p. Секретным ключом пользователя является упорядоченный набор $(x_0, x_1, \ldots, x_i, \ldots, x_{z-1})$ случайных значений $x_i < q$, $i = 1, 2,..., z$.

Формирование цифровой подписи зададим как формирование ответа на случайный запрос, задаваемый значением документа и фиксатора (разового открытого ключа). Это сделает значение запроса от документа неизвестным до шага формирования фиксатора. В качестве битовой цепочки запроса можно взять значение некоторой специфицированной хэш-функции, вычисляемое от подписываемого документа с присоединенным к нему разовым открытым ключом. В этом случае потенциальный нарушитель не сможет заранее, т.е. до формирования значения R, установить значение запроса от документа (в последнем случае подделка подписи становилась бы возможной).

Генерация цифровой подписи владельцем открытого ключа заключается в выполнении следующих шагов.

1. Подписывающий генерирует разовый секрет k, вычисляет значение фиксатора $R = \alpha^k \bmod p$, которое является первым (рандомизирующим) элементом ЭЦП. Затем он объединяет фиксатор и подписываемое сообщение M и вычисляет значение хэш-функции $E = F_H(R, M) = (e_0, e_1, e_2, ..., e_{z-1})$, которое рассматривается как упорядоченная последовательность однобитовых запросов, задаваемых подписываемым документом. (При выполнении процедуры проверки подписи эта битовая цепочка запросов восстанавливается, после чего будет проверяться правильность ответа на нее.)

2. Второй элемент цифровой подписи вычисляется в виде значения S по формуле

$$S = k + \sum_{i=1}^{m} x_i e_i \bmod q..$$

Процедура проверки цифровой подписи (R, S) к сообщению M включает следующие шаги:

1. Вычисляется значение хэш-функции от первого элемента подписи с присоединенным к нему сообщением M: $E = F_H(R, M) = (e_0, e_1, e_2, ..., e_i, ..., e_{z-1})$..

2. Проверяется справедливость следующего сравнения

$$\alpha^S \equiv R \prod_{i=1}^{z} y_i^{e_i} \bmod p. \qquad (6)$$

Если последнее сравнение выполняется, то подпись признается подлинной, в противном случае подпись отвергается.

Уменьшение размера подписи можно достигнуть следующим модифицированием последней схемы цифровой подписи: вместо значения фиксатора R в качестве первого элемента подписи укажем значение запроса E. Тогда значение фиксатора можно вычислить из проверочного уравнения (6) и, если вычисленное значение фиксатора $\tilde{R}$ окажется равным истинному значению фиксатора $R = \alpha^k \bmod p$, то подпись следует признать подлинной. Выполнимость равенства $\tilde{R} = R$ может быть проверена как выполнимость равенства следующих двух значений хэш-функции $E = F_H(R, M)$ и $\tilde{E} = F_H(\tilde{R}, M)$. Вероятность того, что при выполнении равенства $\tilde{E} = E$ имеет место неравенство $\tilde{R} \neq R$, пренебрежимо мала для значений $z \geq 80$ бит. Введение этих изменений проводит к следующей схеме цифровой подписи.

Процедура генерации цифровой подписи к сообщению M.

1. Подписывающий генерирует разовый секрет k, вычисляет значение фиксатора $R = \alpha^k \bmod p$ (разовый открытый ключ). Затем он объединяет фиксатор и подписываемое

сообщение M и вычисляет значение хэш-функции $E = F_H(R, M) = (e_0, e_1, e_2, ..., e_{z-1})$, которое рассматривается как упорядоченная последовательность однобитовых запросов, задаваемых подписываемым документом. Значение E является первым элементом цифровой подписи.

2. Второй элемент цифровой подписи вычисляется в виде значения S по формуле

$$S = k + \sum_{i=0}^{z-1} x_i e_i \bmod q. \tag{7}$$

Процедура проверки подлинности подписи (E, S) к сообщению M включает следующие шаги:

1. Вычисляется значение

$$\tilde{R} = \alpha^S \prod_{i=0}^{z-1} y_i^{-e_i} \bmod p. \tag{8}$$

2. Вычисляется значение хэш-функции от значения $\tilde{R}$ первого элемента подписи с присоединенным к нему сообщением M: $\tilde{E} = F_H(\tilde{R}, M)$.

3. Проверяется выполнимость равенства $\tilde{E} = E$. Если последнее равенство выполняется, то подпись признается подлинной. В противном случае подпись отвергается.

Недостатком последних двух схем цифровой подписи является достаточно большой размер открытого и секретного ключа. Устранение этого недостатка можно добиться на основе следующей идеи. Генерируется секретный ключ x и вычисляется открытый ключ $y = \alpha^x \bmod p$. Секретный ключевой массив (x_0, x_1, …, x_i, …, x_{z-1}) вычисляется при формировании цифровой подписи в соответствии с формулой

$$x_i = 2^i x \bmod n,\ i = 0,1,2,..., z-1.$$

Открытый ключевой массив (y_0, y_1, …, y_i, …, y_{z-1}) вычисляется при проверке подлинности цифровой подписи в соответствии с формулой

$$y_i = y^{2^i} \bmod n,\ i = 0,1,2,..., z-1.$$

Последний массив может быть вычислен также и по формуле

$$y_i = \alpha^{x_i} \bmod n,\ i = 0,1,2,..., z-1.$$

Приняв такое соглашение о расширение секретного и открытого ключа, мы можем показать, что следующая схема выводится из предыдущей:

Процедура генерации цифровой подписи к сообщению M.

1. Подписывающий генерирует разовый секрет k, вычисляет значение фиксатора $R = \alpha^k \bmod p$ (разовый открытый ключ). Затем он объединяет фиксатор и подписываемое сообщение M и вычисляет значение хэш-функции $E = F_H(R, M)$, которое является первым элементом подписи.

2. Второй элемент цифровой подписи вычисляется в виде значения S по формуле

$$S = k + xE \bmod q. \tag{9}$$

Процедура проверки подлинности подписи (*E, S*) к сообщению *M* включает следующие шаги:

1. Вычисляется значение

$$\tilde{R} = y^{-E}\alpha^{S} \bmod p. \qquad (10)$$

2. Вычисляется значение хэш-функции от значения $\tilde{R}$ первого элемента подписи с присоединенным к нему сообщением *M*: $\tilde{E} = F_H\left(\tilde{R}, M\right)$.

3. Проверяется выполнимость равенства $\tilde{E} = E$. Если последнее равенство выполняется, то подпись признается подлинной. В противном случае подпись отвергается.

Уравнение (9), используемое для вычисления второго элемента подписи, фактически реализует вычисления, задаваемые формулой (7), при использование указанного ранее расширения секретного ключа *x*:

$$S \equiv k + \sum_{i=0}^{z-1} x_i e_i \equiv k + \sum_{i=0}^{z-1} 2^i x e_i \equiv k + x\sum_{i=0}^{z-1} 2^i e_i = k + xE \bmod q.$$

Уравнение (10), используемое для вычисления второго элемента подписи, фактически реализует вычисления, задаваемые формулой (8), при использование указанного ранее расширения открытого ключа *y*:

$$\tilde{R} \equiv \alpha^{S}\prod_{i=0}^{z-1} y_i^{-e_i} \equiv \alpha^{S}\prod_{i=0}^{z-1} y^{-2^i e_i} \equiv \alpha^{S} y^{-\sum_{i=0}^{z-1} 2^i e_i} \equiv y^{-E}\alpha^{S} \bmod p.$$

Если в последней схеме цифровой подписи формулу вычисления первого элемента подписи $E = F_H\left(R, M\right)$ заменить на $E = F_H\left(M, R\right)$, то в точности получаем схему Шнорра [15]. Расположение фиксатора *R* перед подписываемым сообщением представляется предпочтительным, поскольку это обеспечит защиту от атак на основе заранее подготовленных пар сообщений с одинаковыми значениями хэш-функций, т.е. от атак, основанных на заранее заготовленных коллизиях.

7.4 Формальное доказательство стойкости схем цифровой подписи

Тот факт, что схема цифровой подписи Шнорра [15] является реализацией протокола с нулевым разглашением секрета с участием подписываемого документа (роль проверяющего может быть выполнена любым лицом, заинтересованным в проверке подлинности подписи) может быть рассмотрено как формальное доказательство стойкости схемы [15]. Действительно, цифровая подпись не содержит никакой новой информации о секретном ключе также как и пары значений фиксатор–ответ в (базовом) протоколе с нулевым разглашением, из которого была выведена схема [15]. Предположение о возможности подделки цифровой подписи означает предположение о возможности нарушителем сформировать правильный ответ в протоколе с нулевым разглашением. Таким образом, схема цифровой подписи обладает стойкостью одного порядка со стойкостью базового протокола с нулевым разглашением.

На самом деле, вопрос о доказательстве нулевого разглашения секрета в ходе протокола с нулевым разглашением и доказательство практической (вычислительной) невозможности дать правильный ответ на случайный запрос без знания секретного ключа – это разные аспекты оценки стойкости. Покажем, что в протоколе, из которого была выведена схема [15], подделка подписи имеет вычислительную сложность одного порядка с вычислительной сложностью задачи дискретного логарифмирования по простому модулю. Для рассмотрения этого вопроса запишем базовый алгоритм с нулевым разглашением секрета, в котором используется открытый ключ *y*, вычисляемый по

секретному ключу $x < q$ по формуле $y = \alpha^x \bmod p$, где α - число достаточно большого простого порядка q по модулю p:

1. Доказывающий выбирает случайное число $k < q$ (разовый секрет), вычисляет значение фиксатора $R = \alpha^k \bmod p$.

2. Проверяющий генерирует свой случайный запрос в виде z-разрядного двоичного числа E и направляет его доказывающему.

3. Доказывающий вычисляет значение ответа на запрос в виде числа $w = k + xE \bmod q$ и направляет значение w проверяющему.

Проверяющий считает полученный им ответ правильным, а доказывающего подлинным, если выполняется следующее проверочное соотношение

$$\alpha^w = Ry^E \bmod p.. \tag{11}$$

Используя толкование нулевого разглашения в терминах статистической равнозначности равновероятных случайных значений (R, E, w), легко показать, что эти три шага задают протокол с нулевым разглашением секрета. Допустим, что нарушитель знает некоторый алгоритм, который позволяет ему подделать правильный ответ. Тогда он генерирует случайное число $k < q$, вычисляет значение фиксатора $R = \alpha^k \bmod p$. Затем он генерирует два случайных запроса E_1 и E_2 и вычисляет для каждого из них ответ w_1 и w_2. Из формулы (11) можно записать следующее

$$\alpha^{w_1 - w_2} \equiv y^{E_1 - E_2} \equiv \alpha^{x(E_1 - E_2)} \bmod p \;\Leftrightarrow\; w_1 - w_2 \equiv x(E_1 - E_2) \bmod q \;\Rightarrow$$

$$\Rightarrow x = \frac{w_1 - w_2}{E_1 - E_2} \bmod q.$$

(12)

То есть, значение дискретного логарифма x при основании α по модулю p вычислено. Вычисления по формулам (11) вносят несущественный вклад в трудоемкость подделки вычисления значения x, поэтому сложность вычисления дискретного логарифмирования таким способом в два раза больше вычислительной сложности алгоритма поддлеки ответа в протоколе с нулевым разглашением. Это показывает, что любой вычислительно эффективный алгоритм подделки подписи может быть использован как алгоритм эффективного дискретного логарифмирования. Переходя от этого базового протокола к схеме цифровой подписи [15], которая непосредственно выводится из него, можем утверждать, что трудоемкость подделки подписи в схеме [15] и задаче дискретного логарифмирования по простому модулю p (при основании α) имеют одинаковый порядок. Это завершает формальное доказательство стойкости схемы цифровой подписи Шнорра, предложенному в работе [16].

Известный ранее способ [17] формального доказательства стойкости этой схемы подписи представляется более громоздким и в нем также используется предположение о том, что в схеме цифровой подписи используется стойкая хэш-функция, поэтому атакующий не может воспользоваться наличием ее слабостей. Это было первое в мире формальное доказательство стойкости схемы ЭЦП, основанное на вычислительной трудности задачи дискретного логарифмирования, однако оно не может быть распространено на стандарты цифровой подписи России и США [18] из-за отличий в процедуре формирования подписи.

Формальное доказательство стойкости схем цифровой подписи путем их вывода из протоколов с нулевым разглашением, что предложено в работе [16], представляется более общим подходом, и может быть применено для случая указанных стандартов. Однако,

для каждого из них, не в полной мере проработан вопрос об обосновании базового протокола с нулевым разглашением, из которого выводится схема.

Раздел 8. Снижение сложности вычислений в криптосхемах, основанных на задаче дискретного логарифмирования

В алгоритмах и протоколах с открытым ключом наиболее широко применяются операции модульного возведения в большую степень и модульного умножения. Последняя, из этих двух операций, является определяющей, поскольку возведение в степень реализуется с ее использованием. Модульное умножение, в общем случае, выполняется как арифметическое умножение операндов и последующее арифметическое деление результата предыдущей операции на модуль. Результатом модульного умножения является остаток, получаемый при арифметическом делении. В качестве операндов и модуля выступают числа или многочлены, в частности двоичные многочлены как наиболее интересный практический случай. При этом, чаще всего, в качестве модуля используется простое или трудно разложимое число и неприводимый многочлен.

Ускорение модульного умножения может быть достигнуто следующими двумя путями: 1. использованием метода Монтгомери и 2. выбором такого модуля, при котором модульное умножение может быть выполнено без выполнения арифметического деления (путем выполнения арифметического умножения и нескольких операций сложения). Первый способ является общим и может быть применен для любых модулей. Второй способ дает более значительное ускорение модульного умножения, однако он не может быть применен в случаях криптосхем, в которых факторизация модуля является секретной.

8.1. Метод Монтгомери

Рассмотрим применение метода Монтгомери в случае модульного умножения многочленов, заданных над конечным простым полем $GF(p)$. Случай модульного умножения чисел по методу Монтгомери рассмотрен в [5]. Следует отметить, что ускорение операции модульного умножения достигается в случае, когда требуется выполнить достаточно много операций умножения по заданному модулю, что имеет место при выполнении операции возведения в большую степень. Последнее является типичным для криптосхем с открытым ключом, основанных на трудности задачи дискретного логарифмирования в конечной циклической группе и задачи факторизации. Это определяет значимость метода Монтгомери при решении задачи повышения производительности криптографических алгоритмов и протоколов. В рассматриваемом методе каждому многочлену $\mu(x)$ ставится в соответствие некоторый образ-многочлен $F[\mu(x)]$ и вычисление осуществляется над образами, а после получения результата осуществляется переход от образа к многочлену. Предположим, требуется выполнять многократные умножения по модулю многочлена $\eta(x)$. При этом выбирается некоторый многочлен $\rho(x)$, такой что $\text{НОД}(\eta(x),\rho(x))=1$ и $\deg\rho(x)>\deg\eta(x)$, где deg обозначает степень многочлена. Ускорение операции умножения по модулю $\eta(x)$ достигается за счет того, что операция деления на $\eta(x)$ в арифметике Монтгомери заменяется на операцию деления на многочлен $\rho(x)$, в качестве которого выбирается одночлен x^n, где n – натуральное число, и тогда операция деления сводится к записи коэффициентов делимого многочлена, стоящих при степенях переменной, не превосходящих n.

Рассмотрим отображение по Монтгомери. Многочленам $\mu(x)$ и $\gamma(x)$ ставятся в соответствие многочлены $\mu'(x)$ и $\gamma'(x)$, такие что

$$\mu'(x) = \mu(x)\rho(x) \bmod \eta(x) \quad \text{и} \quad \gamma'(x) = \gamma'(x)\rho(x) \bmod \eta(x).$$

Тогда сумме $\mu(x) + \gamma(x)$ соответствует сумма $\mu'(x) + \gamma'(x) = (\mu(x) + \gamma(x))\rho(x) \bmod \eta(x)$, а произведению $\mu(x)\gamma(x) \bmod \eta(x)$ – многочлен $F[\mu(x)\gamma(x)] = \mu'(x)\gamma'(x)\rho^{-1}(x) \bmod \eta(x)$, где $\rho^{-1}(x)$ – многочлен, обратный по модулю $\eta(x)$ к многочлену $\rho(x)$. При этом существует вычислительно эффективная процедура вычисления $F[\mu(x)\gamma(x)]$ по $F[\mu(x)] = \mu'(x)$ и

$F[\gamma(x)] = \gamma'(x)$. Обоснование этой процедуры состоит в следующем. Представим наибольший общий делитель многочленов $\eta(x)$ и $\rho(x)$ в виде их линейной комбинации:

$$\text{НОД}(\eta(x),\rho(x)) = \left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) + \left(\rho^{-1}(x) \bmod \eta(x)\right)\rho(x) = 1, \quad (13)$$

где коэффициенты линейного представления, т.е. многочлены $\eta^{-1}(x) \bmod \rho(x)$ и $\rho^{-1}(x) \bmod \eta(x)$ легко вычисляются, применяя расширенный алгоритма Евклида. Умножая обе части выражения (13) на $\mu'(x)\gamma'(x)$, получаем следующее выражение

$$\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) + \mu'(x)\gamma'(x)\left(\rho^{-1}(x) \bmod \eta(x)\right)\rho(x) = \mu'(x)\gamma'(x),$$

из которого вытекает

$$\begin{aligned} &\mu'(x)\gamma'(x)\left(\rho^{-1}(x) \bmod \eta(x)\right)\rho(x) = \\ &= \mu'(x)\gamma'(x) - \mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x). \end{aligned} \quad (14)$$

Выражение (14) представляет собой равенство многочленов, поэтому левая часть (14) сравнима с правой частью (14) по модулю, равному любому многочлену, в том числе и $\rho(x)\eta(x)$. Записывая (14) по модулю $\rho(x)\eta(x)$, получаем:

$$\begin{aligned} &\mu'(x)\gamma'(x)\left(\rho^{-1}(\mathrm{x}) \bmod \eta(x)\right)\rho(x) \equiv \\ &\equiv \mu'(x)\gamma'(x) - \mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) \bmod \rho(x)\eta(x). \end{aligned} \quad (15)$$

Записывая (14) по модулю $\rho(x)$, получаем:

$$0 \equiv \mu'(x)\gamma'(x) - \mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) \bmod \rho(x). \quad (16)$$

Справедливость сравнения (16) означает, что многочлен, задаваемый как разность $\mu'(x)\gamma'(x) - \mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x)$, делится без остатка на многочлен $\rho(x)$, т.е. модуль, левая и правая часть сравнения (15) делятся без остатка на многочлен $\rho(x)$. В силу последнего сравнение (15) можно переписать в следующем виде:

$$\begin{aligned} &\mu'(x)\gamma'(x)\left(\rho^{-1}(\mathrm{x}) \bmod \eta(x)\right) \equiv \\ &\equiv \frac{\mu'(x)\gamma'(x) - \mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x)}{\rho(x)} \bmod \eta(x). \end{aligned} \quad (17)$$

Рассмотрим многочлен $\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x)$ в правой части формулы (17). Представим произведение $\mu'(x)\gamma'(x)$ в виде $\mu'(x)\gamma'(x) = q(x)\rho(x) + \left(\mu'(x)\gamma'(x) \bmod \rho(x)\right)$, где $q(x)$ – частное от деления многочлена, равного произведению $\mu'(x)\gamma'(x)$, на многочлен $\rho(x)$. Тогда получаем

$$\begin{aligned} &\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) = \\ &= q(x)\rho(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) + \left(\mu'(x)\gamma'(x) \bmod \rho(x)\right)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) = \\ &= q(x)\rho(x)\left(\eta^{-1}(x) \bmod \rho(x)\right)\eta(x) + \\ &+ q^{*}(x)\rho(x)\eta(x) + \left(\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right) \bmod \rho(x)\right)\eta(x) = \\ &= q^{**}(x)\rho(x)\eta(x) + \left(\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right) \bmod \rho(x)\right)\eta(x), \end{aligned} \quad (18)$$

где $q^{*}(x)$ – частное от деления многочлена $\left(\mu'(x)\gamma'(x) \bmod \rho(x)\right)\left(\eta^{-1}(x) \bmod \rho(x)\right)$ на многочлен $\rho(x)$; $q^{**}(x) = q(x)\left(\eta^{-1}(x) \bmod \rho(x)\right) + q^{*}(x)$. С учетом (18) выражение (17) переписываем следующим образом:

$$\mu'(x) \circ \gamma'(x) = \mu'(x)\gamma'(x)\left(\rho^{-1}(x) \bmod \eta(x)\right) \equiv$$
$$\equiv \frac{\mu'(x)\gamma'(x) - \left(\mu'(x)\gamma'(x)\left(\eta^{-1}(x) \bmod \rho(x)\right) \bmod \rho(x)\right)\eta(x)}{\rho(x)} \bmod \eta(x), \qquad (19)$$

где многочлен, находящийся в числителе дроби, делится без остатка на многочлен $\rho(x)$. Последнее соотношение задает некоторую операцию (°) над $\mu'(x)$ и $\gamma'(x)$, которая дает образ произведения $\mu(x)$ и $\gamma(x)$. При этом операция ° реализуется как процедура вычисления правой части (19), где фактически не требуется выполнение деления на модуль $\eta(x)$. Действительно, нахождение частного от деления некоторого многочлена $\alpha(x)$ на многочлен $\rho(x) = x^n$ и взятие остатка от этого деления сводится к выписыванию коэффициентов $\alpha(x)$. При этом степень многочлена, находящегося в числителе дроби в правой части (19) не превышает значения $\deg \rho(x) - 1 + \deg \eta(x)$, поэтому степень самой дроби не превышает значения $\deg \eta(x) - 1$.

Возведение многочлена $\alpha(x)$ в степень k выполняется как вычисление образа многочлена $\alpha^k(x)$, которым является многочлен $F\left[\alpha^k(x)\right]$, и переходом от образа к прообразу (редукция Монтгомери). Вычисление образа $F\left[\alpha^k(x)\right]$ выполняется по формуле

$$F\left[\alpha^k(x)\right] = \underbrace{F\left[\alpha(x)\right] \circ F\left[\alpha(x)\right] \circ \ldots \circ F\left[\alpha(x)\right]}_{k \text{ раз}} = F^{(k)}\left[\alpha(x)\right],$$

где возведение в степень (n) выполняется с использованием алгоритма быстрого возведения в степень (см., например, с. 55 в [5]), основанном на последовательном возведении в квадрат:

$$F^{(2^{i+1})}\left[\alpha(x)\right] = F^{(2^i)}\left[\alpha(x)\right] \circ F^{(2^i)}\left[\alpha(x)\right], \quad i = 0,1,2 \ldots \ .$$

Для нахождения прообраза используется формула

$$\alpha^k(x) = F\left[\alpha^k(x)\right]\rho^{-1}(x) \bmod \eta(x).$$

8.2. Выбор простых чисел специального вида

Вычислительная сложность модульного умножения чисел существенно зависит от вида числа используемого в качестве модуля. В криптосхемах в качестве модуля чаще всего используются простые числа, поэтому будем рассматривать случай простого модуля, однако способ ускорения операции модульного умножения, который состоит в выборе модуля, запись которого в некоторой системе счисления содержит малое число ненулевых разрядов, применим также и в случае, когда модуль не является простым числом. Наиболее простой для рассмотрения случай относится к двоичной системе счисления.

Рассмотрим модуль, представляющий собой число вида $p = 2^n \pm 1$, и два произвольных числа $a < p$ и $b < p$. Арифметическое произведение ab равно некоторому числу w, которое можно выразить в виде

$$w = ab = w_1 \| w_2 = w_1 2^n + w_2,$$

где числа w_1 и w_2 имеют разрядность не более $n + 1$ и n бит, соответственно; знак || обозначает операцию конкатенации (присоединения) двух битовых строк. Далее запишем

$$ab = w_1 2^n \pm w_1 + w_2 \mp w_1 = w_1\left(2^n \pm 1\right) + w_2 \mp w_1.$$

Из последнего соотношения имеем сравнение

$$ab \equiv w_2 \mp w_1 \bmod p.$$

Таким образом, результат умножения чисел $a < p$ и $b < p$ по модулю может быть получен без выполнения операции арифметического деления. Достаточно выполнить одно арифметическое умножение и одну или две операции сложения (может оказаться, что значение $w_2 \mp w_1 > p$ и тогда из него потребуется вычесть число p).

Только для сравнительно редких натуральных чисел n значение $p = 2^n \pm 1$ является простым. Для любого значения n можно найти простые числа вида

$$p = 2^n \pm 2^k \pm 2^h \pm 2^g \pm 1, \tag{20}$$

где $n/2 - 1 > k > h > g > 0$. При использовании чисел такого вида в качестве модуля модульное умножение также может быть выполнено без применения операции арифметического деления. Достаточно выполнить арифметическое умножение и несколько сложений. По аналогии с рассмотренным ранее случаем для произведения чисел $a < p$ и $b < p$ можем записать

$$\begin{aligned} ab = w = w_1 2^n + w_2 = w_1 2^n \pm w_1 2^k \pm w_1 2^h \pm w_1 2^g \pm w_1 + w_2 \mp w_1 2^k \mp \\ \mp w_1 2^h \mp w_1 2^g \mp w_1 = w_1\left(2^n \pm 2^k \pm 2^h \pm 2^g \pm 1\right) + \\ + w_2 \mp w_1 2^k \mp w_1 2^h \mp w_1 2^g \mp w_1. \end{aligned} \tag{21}$$

Следовательно,

$$ab \equiv w_2 \mp w_1 2^k \mp w_1 2^h \mp w_1 2^g \mp w_1 \bmod p. \tag{22}$$

В правой части последнего сравнения находится число, разрядность которого не превышает значение $n + k + 2$. Обозначим это число как

$$W = W_1 \| W_2 = W_1 2^n + W_2,$$

где числа W_1 и W_2 имеют разрядность не более $k + 2$ и n бит. Аналогично (22) получаем следующие соотношения:

$$W = W_1\left(2^n \pm 2^k \pm 2^h \pm 2^g \pm 1\right) + W_2 \mp W_1 2^k \mp W_1 2^h \mp W_1 2^g \mp W_1, \tag{23}$$

$$ab \equiv W \equiv W_2 \mp W_1 2^k \mp W_1 2^h \mp W_1 2^g \mp w_1 \bmod p. \tag{24}$$

число W_2 имеет разрядность не более n бит, а число $2^k W_1$ имеет разрядность не более $2k + 2 < n$ бит.

Таким образом, при модуле вида (22) модульное умножение выполняется без выполнения операции арифметического деления путем выполнения арифметического умножения операндов, от 8 до 10 операций сложения и 6 операций арифметического сдвига битовых строк (умножение на 2^k, 2^h и 2^g).

Для модуля вида

$$p = 2^n \pm 2^k \pm 2^h \pm 1, \tag{25}$$

где $n/2 - 1 > k > h > 0$, получаем следующие формулы для вычисления модульного умножения чисел $a < p$ и $b < p$:

$$W = W_1 \| W_2 = w_2 \mp w_1 2^k \mp w_1 2^h \mp w_1; \tag{26}$$

$$ab = w = w_1 \| w_2 \equiv W_2 \mp W_1 2^k \mp W_1 2^h \mp w_1 \bmod p. \tag{27}$$

Более интересным для практики случаем является модуль вида

$$p = 2^n \pm 2^k \pm 1, \tag{28}$$

где $n/2 - 1 > k > 0$, для которого имеем следующие формулы для вычисления модульного умножения чисел $a < p$ и $b < p$:

$$ab = w = w_1 \| w_2 = w_1 2^n + w_2;$$

$$W = W_1 \| W_2 = w_2 \mp w_1 2^k \mp w_1; \tag{29}$$

$$ab \equiv W_2 \mp W_1 2^k \mp W_1 \bmod p. \tag{30}$$

Аналогично тому, как это сделано для случая записи модуля в двоичном виде, для любой системы счисления существует вид модуля, при котором можно предложить процедуру вычисления результата операции модульного умножения, которая свободна от выполнения операции деления. Например, для этой цели интерес представляют модули-числа, имеющие в T-ичной системе счисления вид

$$p = T^n \pm T^k \pm d_h T^h \pm d_0, \tag{31}$$

где $n/2 - 1 > k > h > 0$; $1 \le d_h < T$; $1 \le d_0 < T$; d_h и d_0 – значения цифр в нулевом и h-ом разряде T-ичной записи числа p. Рассмотрим значение результата w арифметического умножения двух чисел $a < p$ и $b < p$:

$$ab = w = w_1 \| w_2 = w_1 T^n + w_2,$$

где T-ичное число w_1 имеет длину не более $n + 1$ знаков, причем в старшем разряде числа w_1 записана единица; T-ичное число w_2 имеет длину не более n знаков. С учетом представления модуля (31) можно записать

$$ab = w_1\left(T^n \pm T^k \pm d_h T^h \pm d_0\right) + w_2 \mp w_1 T^k \pm w_1 d_h T^h \pm w_1 d_0;$$
$$ab \equiv w_2 \mp w_1 T^k \pm w_1 d_h T^h \pm w_1 d_0 \bmod p.$$

В правой части последнего сравнения находится число W длины $k + n + 1$ разрядов, которое представим в виде

$$W = W_1 \| W_2 = W_1 T^n + W_2 =$$
$$= W_1\left(T^n \pm T^k \pm d_h T^h \pm d_0\right) + W_2 \mp W_1 T^k \pm W_1 d_h T^h \pm W_1 d_0;$$

$$ab \equiv W \equiv W_2 \mp W_1 T^k \pm W_1 d_h T^h \pm W_1 d_0 \bmod p. \tag{32}$$

В последних соотношениях разрядность числа W_1 равна $k + 1$ знаков, разрядность слагаемого $W_1 T^k$ равна $2k + 1$ знаков, причем $2k + 1 < n$ (в силу условия $n/2 - 1 > k$, относящегося к (31)). Длина числа W_2 не более n знаков, а разрядность всей правой части последнего сравнения не превосходит значения $n + 1$ знаков. Поэтому вычисление правой части (32) даст число, меньшее, чем p. С некоторой, достаточно малой, вероятностью при выборе случайных множителей a и b можно получить значение этой правой части,

превосходящие модуль *p*. В этом случае из значения правой части (32) потребуется вычесть значение модуля.

8.3. Выбор неприводимых многочленов специального вида

Повышение производительности двухключевых криптосхем, построенных с использованием трудности задачи дискретного логарифмирования в конечных полях многочленов, заданных над простым полем $GF(p)$, связано со снижением сложности операции умножения многочленов по модулю неприводимого многочлена. Прямолинейная реализация этой операции связана с выполнением арифметического умножения двух многочленов и арифметическим деление полученного результата на заданный в качестве модуля неприводимый многочлен. В качестве результата модульного умножения берется остаток от указанной операции деления. Остатком является многочлен, степень которого, по крайней мере, на единицу меньше степени многочлена-модуля. При этом, временная сложность операции арифметического деления существенно превышает сложность операции арифметического умножения многочленов.

Уменьшение вычислительной сложности операции модульного умножения многочленов может быть достигнуто следующими способами:

-заменой деления на заданный модуль-многочлен степени n на деление на одночлен степени $n + 1$ в соответствии с методом Монтгомери;

-выбором в качестве модуля многочлена с небольшим числом ненулевых коэффициентов.

Рассмотрим второй способ. Возможность его применения связана с тем, что при построении криптосхем, в которых применяются вычисления в полях $GF(p^n)$, элементы которых заданы в виде многочленов степени менее n, важным вопросом является делимость порядка мультипликативной группы таких полей на достаточно большое простое число. При этом порядок мультипликативной группы равен $p^n - 1$ и не зависит от выбора конкретного неприводимого многочлена, задаваемого как модуль операции модульного умножения. Для заданных значений p и n можно выбрать в качестве модуля многочлен с минимальным числом ненулевых коэффициентов. Например, для случая простых значений степени n и делимости значения $p - 1$ на n существуют неприводимые двухчленны вида $\eta(x) = x^n + c_0$, где $c_0 \in GF(p)$.

Пусть требуется перемножить многочлены $\mu(x)$ и $\gamma(x)$, степени которых равны $n - 1$, по модулю $\eta(x) = x^n + c_0$. Выполняя арифметическое умножение $\mu(x)$ и $\gamma(x)$, получим многочлен $\omega(x)$ степени $2n - 2$, который можно представить в виде

$$\omega(x) = x^n \omega_1(x) + \omega_2(x),$$

где $\omega_1(x)$ –-многочлен степени $n - 2$; $\omega_2(x)$ –-многочлен, степень которого не превышает значения $n - 1$. С учетом такого представления $\omega(x)$ имеем

$$\begin{aligned}\mu(x)\gamma(x) = \omega(x) &= x^n \omega_1(x) + c_0\omega_1(x) - c_0\omega_1(x) + \omega_2(x) = \\ &= \omega_1(x)\left(x^n + c_0\right) + \omega_2(x) - c_0\omega_1(x) = \\ &= \omega_1(x)\eta(x) + \omega_2(x) - c_0\omega_1(x).\end{aligned}$$

Равные многочлены сравнимы по любому модулю, в том числе и по модулю $\eta(x)$, поэтому получаем

$$\mu(x)\gamma(x) \equiv \omega_2(x) - c_0\omega_1(x) \bmod \eta(x).$$

В последнем выражении многочлен $\omega_2(x)$ получается простым выписыванием $n-1$ коэффициента многочлена $\omega(x)$ при старших степенях (от $2n-2$ до n) переменной x. Многочлен $\omega_1(x)$ представляет собой часть многочлена $\omega(x)$. Умножение $\omega_1(x)$ на коэффициент c_0 требует выполнения не более n дополнительных умножений в поле $GF(p)$ к $\approx n^2$, выполняемых при арифметическом умножении многочленов $\mu(x)$ и $\gamma(x)$. Операцией вычитания (сложения) многочленов можно пренебречь. Таким образом, умножение многочленов по модулю неприводимого двухчлена выполняется без выполнения операции деления на модуль и имеет трудоемкость, равную $\approx n^2$ умножений в конечном поле, над которым заданы многочлены.

Для реализации криптографических алгоритмов и протоколов особый интерес представляет использование двоичных многочленов. Это связано с тем, что любой возможный n-битовый блок данных может трактоваться (интерпретироваться) как элемент двоичного поля $GF(2^s)$, заданный упорядоченным набором коэффициентов двоичного многочлена. При этом различным блокам данных соответствуют различные элементы поля $GF(2^s)$, т.е. имеет место взаимно однозначное соответствие множеств возможных значений n-битовых блоков данных и двоичных многочленов степеней $z < s$. В случае двоичных полей $GF(2^n)$, т.е. полей двоичных многочленов, минимальное число ненулевых коэффициентов в неприводимом многочлене равно трем. В этом случае имеем $\eta(x) = x^n + x^k + 1$.

Пусть требуется перемножить двоичные многочлены $\mu(x)$ и $\gamma(x)$ степени $n-1$, по модулю $\eta(x) = x^n + x^k + 1$. Результатом арифметического умножения многочленов $\mu(x)$ и $\gamma(x)$ является некоторый многочлен $\omega(x)$, степень которого равна $2n-2$. Аналогично рассмотренному ранее случаю, двоичный многочлен $\omega(x)$ можно представить в виде

$$\begin{aligned}\omega(x) = \mu(x)\gamma(x) &= x^n\omega_1(x) + \omega_2(x) = \\ &= x^n\omega_1(x) + x^k\omega_1(x) - x^k\omega_1(x) + \omega_1(x) - \omega_1(x) + \omega_2(x) = \\ &= \omega_1(x)\left(x^n + x^k + 1\right) + \omega_2(x) - x^k\omega_1(x) - \omega_1(x) = \\ &= \omega_1(x)\eta(x) + \omega_2(x) - x^k\omega_1(x) - \omega_1(x).\end{aligned}$$

где $\omega_1(x)$ и $\omega_2(x)$ – многочлены степени не более $n-2$ и $n-1$, соответственно. Из последней формулы получаем следующее сравнение

$$\mu(x)\gamma(x) \equiv \omega_2(x) - x^k\omega_1(x) - \omega_1(x) \bmod \eta(x),$$

где $x^k\omega_1(x)$ – многочлен, степень которого равна значению $n+k-2$. Теперь рассмотрим многочлен $W(x) = \omega_2(x) - x^k\omega_1(x) - \omega_1(x)$. Легко видеть, что $W(x)$ можно представить в виде

$$W(x) = x^n W_1(x) + W_2(x),$$

где $W_1(x)$ – многочлен степени $k-2$; $W_2(x)$ – многочлен, степень которого не превышает значения $n-1$. С учетом последнего выражения можем записать следующее:

$$\omega_2(x) - x^k\omega_1(x) - \omega_1(x) = W(x) = x^n W_1(x) + W_2(x) =$$
$$= x^n W_1(x) + x^k W_1(x) + W_1(x) + W_2(x) - x^k W_1(x) - W_1(x) =$$
$$= W_1(x)\eta(x) + W_2(x) - x^k W_1(x) - W_1(x) \Rightarrow$$
$$\Rightarrow \quad \omega_2(x) - x^k\omega_1(x) - \omega_1(x) \equiv W_2(x) - x^k W_1(x) - W_1(x) \bmod \eta(x).$$

Следовательно, имеем

$$\mu(x)\gamma(x) \equiv W_2(x) - x^k W_1(x) - W_1(x) \bmod \eta(x),$$

где степень многочлена $x^k W_1(x)$ равна $2k - 2$. Видно, что если взять двоичный трехчлен $\eta(x) = x^n + x^k + 1$ со значение степени $k \le n/2$, то степень многочлена, находящегося в правой части последнего сравнения не превышает значения $n - 1$.

Таким образом, умножение по модулю двоичного трехчлена можно выполнить без выполнения арифметического деления многочленов. Достаточно выполнить одно арифметическое умножение и четыре сложения.

В интересном для практики случае неприводимых двоичных многочленов, степень которых кратна числу 8, минимальное число ненулевых коэффициентов в неприводимом многочлене такого типа равно пяти и они имеют вид

$$\eta(x) = x^n + x^k + x^h + x^g + 1.$$

Выполнение умножения многочлены $\mu(x)$ и $\gamma(x)$ степени $n - 1$ по модулю такого неприводимого многочлена при $n/2 \ge k > h > g > 0$ может быть реализовано как выполнение одной операции арифметического умножения и восьми операций сложения по следующему алгоритму:

1. Вычислить многочлен $\omega(x) = \mu(x)\gamma(x)$ степени $2n - 2$ и представить его в виде

$$\omega(x) = x^n\omega_1(x) + \omega_2(x).$$

2. Вычислить многочлен

$$W(x) = \omega_2(x) - x^k\omega_1(x) - x^h\omega_1(x) - x^g\omega_1(x) - \omega_1(x),$$

степень которого равна $n + k - 2$, и представить его в виде суммы $W(x) = x^n W_1(x) + W_2(x)$.

3. Вычислить многочлен $W_2(x) - x^k W_1(x) - x^h W_1(x) - x^g W_1(x) - W_1(x)$, степень которого не превышает значения $n - 1$ и который равен $\mu(x)\gamma(x) \bmod \eta(x)$.

8.4. Генерация неприводимых многочленов

Для генерации неприводимых многочленов общего вида может быть использован следующий алгоритм:

1. Формируем произвольный многочлен $f(x)$ над полем GF(p).

2. Выполняем его проверку на неприводимость по детерминистическому тесту, который рассматривается далее.

3. Если тест на неприводимость показал, что $f(x)$ является неприводимым, то вывести этот многочлен как результат выполнения алгоритма. В противном случае перейти к шагу 1.

Выход: неприводимый многочлен $f(x)$.

В качестве теста на неприводимость многочлена $f(x)$ степени s, коэффициенты которого лежат в простом поле GF(p), можно использовать следующий алгоритм.

1. Находим все простые делители числа $p^s - 1 = \prod_{i=1}^{z} r_i^{\alpha_i}$ и устанавливаем значение счетчика $c = 1$.

2. Выбираем случайный многочлен $\rho(x)$ над полем GF(p), степень которого меньше значения s: $\rho(x) = \sum_{j=0}^{s-1} k_j x^{\alpha_j}$

3. Проверяем выполнимость сравнения $(\rho(x))^{p^s-1} = 1 \bmod f(x)$ и условия $(\rho(x))^{\frac{p^s-1}{r_i}} \neq 1 \bmod f(x)$ для всех $i \in \{1, 2, ..., z\}$. Если нет, то перейти к шагу 4, если да, то к шагу 5.

4. Если $c < N = 30$, то прирастить счетчик $c = c + 1$ и перейти к шагу 2. Иначе перейти к шагу 6. (Значение N устанавливает сколько различных случайных многочленов $\rho(x)$ будут опробованы для того, чтобы признать текущий многочлен $f(x)$ предположительно приводимым).

5. Многочлен $f(x)$ является неприводимым.

6. Многочлен $f(x)$ предположительно является приводимым.

С некоторой достаточно малой вероятностью данный тест может пропустить некоторый неприводимый многочлен, т.е. признать его предположительно приводимым. Для уменьшения этой вероятности можно увеличить значение параметра N, например до $N = 50$.

Покажем, что многочлен $f(x)$, идентифицируемый данным тестом как неприводимый, действительно является неприводимым. Допустим противное, т.е. то, что $f(x)$ представим в виде произведения неприводимых многочленов меньшей степени, например двух многочленов $\alpha(x)$ и $\beta(x)$, имеющих степени s_1 и s_2, соответственно. Поскольку по предположению $f(x) = \alpha(x)\beta(x)$, то имеет место соотношение $s = s_1 + s_2$. Переход к шагу 5, описанного теста, имеет место только, если будет найден многочлен $\rho(x)$, такой, что имеет место выражение $(\rho(x))^{p^s-1} = 1 \bmod f(x)$ и выполняется условие $(\rho(x))^{\frac{p^s-1}{r_i}} \neq 1 \bmod f(x)$ для всех простых делителей числа $p^s - 1$. По предположению многочлены $\alpha(x)$ и $\beta(x)$ являются неприводимыми, следовательно, для многочлена $\rho(x)$ справедлива система сравнений

$$\begin{cases} (\rho(x))^{p^{s_1}-1} = 1 \bmod \alpha(x) \\ (\rho(x))^{p^{s_2}-1} = 1 \bmod \beta(x) \end{cases}.$$

Возводя первое сравнение в степень $p^{s_2}-1$, а второе – в степень $p^{s_1}-1$, получаем следующую систему

$$\begin{cases}(\rho(x))\left(p^{s_1}-1\right)\left(p^{s_2}-1\right)=1 \bmod \alpha(x) \\ (\rho(x))\left(p^{s_1}-1\right)\left(p^{s_2}-1\right)=1 \bmod \beta(x)\end{cases}.$$

Поскольку наибольший общий делитель многочленов $\alpha(x)$ и $\beta(x)$ равен 1, то из последней системы сравнений вытекает справедливость сравнения

$$(\rho(x))\left(p^{s_1}-1\right)\left(p^{s_2}-1\right)=1 \bmod f(x), \text{ где } f(x)=\alpha(x)\beta(x).$$

Последнее сравнение означает, что порядок многочлена $\rho(x)$ не может быть больше значения $\left(p^{s_1}-1\right)\left(p^{s_2}-1\right)=p^{s_1+s_2}-p^{s_1}-p^{s_2}+1<p^{s_1+s_2}-1=p^{s}-1$ по модулю $f(x)$. Однако, было проверено, что многочлен $\rho(x)$ удовлетворяет условиям шага 3, а следовательно его порядок по модулю $f(x)$ равен $p^{s}-1$. Получено противоречие, которое доказывает, что многочлен $f(x)$ действительно является неприводимым.

В случае генерации двоичных многочленов (p=2), степени которых представимы в виде $s=2^h$ (h – натуральное число) имеем такое разложение для шага 1: $p^{s}-1=2^{2^h}-1=\prod_{k=1}^{h}\left(2^{2^{h-k}}+1\right)$. Это можно использовать для упрощения компьютерной программы для генерации двоичных многочленов, имеющих указанные значения степени. Также и для других частных случаев, когда значения p и s заранее известны, все простые делители числа $p^{s}-1$ можно вычислить и записать в виде таблицы, к которой будет обращаться компьютерная программа, вместо того, чтобы на шаге 1 выполнять процедуру факторизации числа $p^{s}-1$.

Список литературы

1. Молдовян А.А., Молдовян Н.А. Введение в криптосистемы с открытым ключом. – С-Петербург, Петербург–БХВ, 2005. – 288 с.

2. ElGamal T. A public key cryptosystem and a signature scheme based on discrete logarithms // IEEE Transactions on Information Theory. 1985, Vol. IT-31, No. 4. pp.469 – 472.

3. ISO/IEC 9798-5:2009(E) «Information technology — Security techniques — Entity authentication — Part 5: Mechanisms using zero-knowledge techniques».

4. Демьянчук А.А, Молдовян Д.Н., Молдовян А.А. Алгоритмы открытого шифрования в протоколах с нулевым разглашением секрета // Вопросы защиты информации. 2013. № 2. С. 22-27.

5. Молдовян Н.А.Теоретический минимум и алгоритмы цифровой подписи. – С-Петербург, Петербург–БХВ, 2010. – 304 с.

6. Chaum D. Blind Signatures for Untraceable Payments // Advances in Cryptology: Proc. of CRYPTO'82. Plenum Press, 1983. P. 199–203.

7. Bleichenbacher D. Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS #1 // Advances in Cryptology – CRYPTO'98. Springer-Verlag LNCS. 1998. Vol. 1462. P. 1–12.

8. Diffie W., Hellman M.E. New Directions in Cryptography // IEEE Transactions on Information Theory. 1976, Vol. IT-22. pp. 644 – 654.

9. Rabin M.O. Digitalized signatures and public key functions as intractable as factorization. – Technical report MIT/LCS/TR-212, MIT Laboratory for Computer Science, 1979.

10. Молдовян Н.А. Практикум по криптосистемам с открытым ключом. – С-Петербург, Петербург–БХВ, 2007. – 298 с.

11. Crandall R., Pomerance C.: Prime Numbers - A Computational Perspective. – 2nd ed., New York, Springer. 2002. – 604 p.

12. Zierler N. Primitive Trinomials Whose Degree is a Mersenne Exponent // Information and control. 1969. Vol.15. P. 67–69.

13. Демьянчук А.А, Мирин А.Ю., Молдовян Н. А. Типы и приложения протоколов с нулевым разглашением секрета // Информационно-управляющие системы. 2013. № 3. С. 67-73.

14. Fiat A., Shamir A. How to prove yourself: Practical solutions to identification and signature problems // Advances in cryptology – CRYPTO'86, Springer-Verlag LNCS, 1987, vol. 263, pp. 186-194.

15. Schnorr C.P. Efficient signature generation by smart cards // J. Cryptology. 1991. vol. 4. pp. 161-174.

16. Молдовян А.А., Молдовян Д.Н., Васильев И.Н., Головачев Д.А. Протоколы с нулевым разглашением секрета и обоснование безопасности схем цифровой подписи // Вопросы защиты информации. 2011. № 4. С.6-11.

17. D. Pointcheval, and J. Stern. Security Arguments for Digital Signatures and Blind Signatures // Journal of Cryptology. 2000. Vol. 13. P. 361-396.

18. Koblitz N. and Menezes A.J. Another Look at "Provable Security" // Journal of Cryptology. 2007. V. 20. P. 3-38.

19. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си // Триумф, 2002. — 816 с.

УНИВЕРСИТЕТ ИТМО

Миссия университета – генерация передовых знаний, внедрение инновационных разработок и подготовка элитных кадров, способных действовать в условиях быстро меняющегося мира и обеспечивать опережающее развитие науки, технологий и других областей для содействия решению актуальных задач.

КАФЕДРА БЕЗОПАСНЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Кафедра «Безопасные информационные технологии» (БИТ) осуществляет подготовку специалистов по специальности 090103 «Организация и технология защиты информации», бакалавров и магистров по направлению 090900 «Информационная безопасность». Партнерами кафедры являются ОАО «Воентелеком», Санкт-Петербургский институт информатики и автоматизации РАН, Военная академия Генерального штаба ВС РФ, Военно-космическая академия им. А.Ф. Можайского, Комитет по управлению городским имуществом администрации Санкт-Петербурга и другие научные организации и вузы.

На кафедре БИТ осуществляется выполнение научно-исследовательских и опытно-конструкторских работ в области безопасности и защиты информации телекоммуникационных вычислительных систем, включающее:

- Проведение анализа защищенности автоматизированных систем обработки данных и средств вычислительной техники.
- Проектирование и разработка защищенных систем передачи информации.
- Сертификация средств защиты и автоматизированных систем с имеющимися средствами защиты на предмет соответствия определенному классу защищенности.
- Оценка угроз информации и информационных угроз.
- Мониторинг информационных потоков на естественном языке в открытых телекоммуникационных сетях.
- Методы идентификации пользователей в сети Интернет.
- Методы построения систем обнаружения вторжений.
- Методы проектирования криптографических вычислительных систем, устойчивых к современным типам атак.

Выпускники и сотрудники кафедры являются авторами некоторых основополагающих Руководящих Документов ФСТЭК России.

Молдовян А.А., Молдовян Д.Н., Левина А.Б..

Протоколы аутентификации с нулевым разглашением секрета

Учебное пособие

В авторской редакции
Редакционно-издательский отдел Университета ИТМО
Зав. РИО Н.Ф. Гусарова
Подписано к печати
Заказ №
Тираж
Отпечатано на ризографе

Редакционно-издательский отдел
Университета ИТМО
197101, Санкт-Петербург, Кронверкский пр., 49