Глава 21

Схемы идентификации

21.1. Схема Фейге–Фиата–Шамира

Схема цифровой подписи и аутентификации, разработанная Амосом Фиатом (Amos Fiat) и Ади Шамиром (Adi Shamir), обсуждается в [566, 567]. Уриель Фейге (Uriel Feige), Фиат и Шамир модифицировали алгоритм, превратив его в доказательство подлинности с нулевым знанием [544, 545]. Это лучшее доказательство подлинности с нулевым разглашением.

9 июля 1986 г. три автора подали заявку на получение патента США [1427]. Из-за потенциальных военных приложений заявка была рассмотрена оборонным ведомством. Иногда Патентное бюро выдает не патент, а документ, называемый приказом об установлении режима секретности. 6 января 1987 г., за три дня до истечения шестимесячного периода, по просьбе армии Патентное бюро издало такое распоряжение, заявив, что "...раскрытие или публикация предмета заявки ...может причинить ущерб национальной безопасности...". Авторам было приказано уведомить всех граждан США, которые знали о проводимых исследованиях, что несанкционированное разглашение информации может привести к двумя годам тюремного заключения, штрафу в 10000 долларов или тому и другому одновременно. Более того, авторы должны были сообщить комиссару по патентам и торговым знакам обо всех иностранных гражданах, которые получили доступ к этой информации.

Это было абсурдно. В течение второй половины 1986 г. авторы представляли свою работу на конференциях в Израиле, Европе и США. Они даже не были американскими гражданами, а вся работа была выполнена в Институте Вейцмана (Weizmann) в Израиле.

Информация стала распространяться в научном сообществе и прессе. В течение двух дней секретное распоряжение было аннулировано. Шамир и его коллеги считают, что на отмене секретного распоряжения настояло АНБ, хотя оно не давало никаких официальных комментариев. Дальнейшие подробности этой странной истории описаны в [936].

Упрощенная схема идентификации Фейге–Фиата–Шамира

Перед выдачей любых закрытых ключей арбитр выбирает случайный модуль, n, представляющий собой произведение двух больших простых чисел. В реальной жизни длина n должна быть не меньше 512 битов и как можно

ближе к 1024 битам. Модель n может общим для группы претендентов, доказывающих подлинность своих идентификационных данных. (Использование целых чисел Блюма (Blum) упрощает вычисления, но не является необходимым для обеспечения безопасности.)

Для того чтобы сгенерировать открытый и закрытый ключи Пегги, доверенный арбитр выбирает число v, являющееся квадратичным вычетом по модулю n. Другими словами, выбирается число v, такое, что уравнение $x^2 \equiv v \pmod{n}$ имеет решение и существует $v^{-1} \bmod n$. Это число v и является открытым ключом Пегги. Затем вычисляется наименьшее число s, для которого выполняется условие $s \equiv sqrt\left(v^{-1}\right) \pmod{n}$. Это — закрытый ключ Пегги. Используется следующий протокол идентификации.

1. Пегги выбирает случайное число r, меньшее n. Затем она вычисляет число $x = r^2 \bmod n$ и посылает его Виктору.
2. Виктор посылает Пегги случайный бит b.
3. Если $b = 0$, то Пегги посылает Виктору число r. Если $b = 1$, то Пегги посылает Виктору число $y = r \times s \bmod n$.
4. Если $b = 0$, то Виктор проверяет условие $x = r^2 \bmod n$, убеждаясь, что Пегги знает значение sqrt(x). Если $b = 1$, то Виктор проверяет условие $x = y^2 v \bmod n$, убеждаясь, что Пегги знает значение sqrt (v^{-1}).

Это единственный раунд протокола, называемый **аккредитацией**. Пегги и Виктор повторяют его t раз, пока Виктор не убедится, что Пегги знает число s. Протокол относится к категории "разделяй и выбирай". Если Пегги не знает числа s, то она может подбирать разные числа r, чтобы обмануть Виктора, если он пошлет ей нуль или единицу. Она не может сделать одновременно и то и другое. Шансы на то, что ей удастся обмануть Виктора один раз, равны 50%. Вероятность, что ей удастся обмануть его t раз, равна $1/2^t$.

Виктор может атаковать протокол, выдавая себя за Пегги. Он может начать выполнение протокола с другим верификатором, Валерией. На шаге 1 вместо выбора случайного числа r ему останется просто использовать значение, которое Пегги использовала в прошлый раз. Однако вероятность того, что Валерия на шаге 2 выберет то же значение b, которое Виктор использовал в протоколе с Пегги, равна $1/2$. Следовательно, шансы на то, что он обманет Валерию, равны 50%. Вероятность, что ему удастся обмануть ее t раз, равна $1/2^t$.

Для того чтобы этот протокол работал, Пегги никогда не должна использовать число r повторно. В противном случае, если Виктор на шаге 2 пошлет

Пегги другой случайный бит, то получит оба ответа Пегги. Тогда даже по одному из них он сможет вычислить число s, и для Пегги все будет кончено.

Схема идентификации Фейге–Фиата–Шамира

В своих работах [544, 545] Фейге, Фиат и Шамир показали, как параллельная схема может повысить количество аккредитаций на раунд и уменьшить объем взаимодействия между Пегги и Виктором.

Сначала, как и в предыдущем примере, генерируется произведение двух больших простых чисел, n. Для генерации открытого и закрытого ключей Пегги сначала выбираются k разных чисел: $v_1, v_2, \ldots, v_k$, где каждое v_i является квадратичным вычетом по модулю n. Иными словами, числа v_i выбираются так, чтобы уравнение $x^2 \equiv v_i \bmod n$ имело решение и существовало число $v_i^{-1} \bmod n$. Строка $v_1, v_2, \ldots, v_k$ является открытым ключом. Затем вычисляются наименьшие числа s_i, для которых $s_i = sqrt\left(v^{-1}\right) \bmod n$. Строка $s_1, s_2, \ldots, s_k$ является закрытым ключом.

Протокол имеет следующий вид.

1. Пегги выбирает случайное число r, меньшее n. Затем она вычисляет число $x = r^2 \bmod n$ и посылает его Виктору.
2. Виктор посылает Пегги строку из k случайных битов: $b_1, b_2, \ldots, b_k$.
3. Пегги вычисляет число $y = r\left(s_1^{b_1} s_2^{b_2} \ldots s_k^{b_k}\right) \bmod n$. (Она перемножает значения s_i, соответствующие $b_i = 1$. Если первым битом, посланным Виктором, будет единица, то число s_1 войдет в произведение, а если первым битом будет нуль, то нет, и т.д.) Пегги посылает число y Виктору.
4. Виктор проверяет условие $x = y^2\left(v_1^{b_1} v_2^{b_2} \ldots v_k^{b_k}\right) \bmod n$. (Он перемножает значения v_i, основываясь на случайной двоичной строке. Если его первым битом является единица, то число v_1 войдет в произведение, а если первым битом будет нуль, то нет, и т.д.)

Пегги и Виктор повторяют этот протокол t раз, пока Виктор не убедится, что Пегги знает числа $s_1, s_2, \ldots, s_k$.

Вероятность, что Пегги удастся обмануть Виктора t раз, равна $1/2^{kt}$. Авторы рекомендуют использовать вероятность мошенничества $1/2^{20}$ и предлагают значения $k = 5$ и $t = 4$. Если вы страдаете паранойей, то увеличьте эти значения.

Пример

Рассмотрим работу этого протокола на небольших числах.

Если $n = 35$ (два простых множителя — 5 и 7), то возможными квадратичными вычетами являются числа:

1: $x^2 \equiv 1 \pmod{35}$ имеет решения: $x = 1, 6, 29, 34$.
4: $x^2 \equiv 4 \pmod{35}$ имеет решения: $x = 2, 12, 23, 33$.
9: $x^2 \equiv 9 \pmod{35}$ имеет решения: $x = 3, 17, 18, 32$.
11: $x^2 \equiv 11 \pmod{35}$ имеет решения: $x = 9, 16, 19, 26$.
14: $x^2 \equiv 14 \pmod{35}$ имеет решения: $x = 7, 28$.
15: $x^2 \equiv 15 \pmod{35}$ имеет решения: $x = 15, 20$.
16: $x^2 \equiv 16 \pmod{35}$ имеет решения: $x = 4, 11, 24, 31$.
21: $x^2 \equiv 21 \pmod{35}$ имеет решения: $x = 14, 21$.
25: $x^2 \equiv 25 \pmod{35}$ имеет решения: $x = 5, 30$.
29: $x^2 \equiv 29 \pmod{35}$ имеет решения: $x = 8, 13, 22, 27$.
30: $x^2 \equiv 30 \pmod{35}$ имеет решения: $x = 10, 25$.

Обратные значения по модулю 35 и их квадратные корни перечислены в следующей таблице.

v	v^{-1}	$x = \text{sqrt}(v^{-1})$
1	1	1
4	9	3
9	4	2
11	16	4
16	11	9
29	29	8

Обратите внимание, что у чисел 14, 15, 21, 25 и 30 нет обратных значений по модулю 35, поскольку они не являются взаимно простыми с числом 35. Это имеет смысл, поскольку должно быть $(5-1)(7-1)/4$ квадратичных вычетов по модулю 35, взаимно простых с 35: НОД(x, 35) = 1 (см. раздел 11.3).

Итак, Пегги получает открытый ключ, состоящий из $k = 4$ значений: {4, 11, 16, 29}. Соответствующим закрытым ключом является {3, 4, 9, 8}. Рассмотрим один раунд протокола.

1. Пегги выбирает случайное число $r = 16$, вычисляет $16^2 \bmod 35 = 11$ и посылает его Виктору.
2. Виктор посылает Пегги строку случайных битов: {1, 1, 0, 1}
3. Пегги вычисляет число $16\left(3^1 \times 4^1 \times 9^0 \times 8^1\right) \bmod 35 = 31$ и посылает его Виктору.
4. Виктор проверяет условие $31^2\left(4^1 \times 11^1 \times 16^1 \times 29^1\right) \bmod 35 = 11$.

Пегги и Виктор повторяют этот протокол t раз, каждый раз с новым случайным числом r, пока Виктор не будет удовлетворен.

Небольшие числа, подобные использованным в примере, не обеспечивают реальной безопасности. Но если длина числа n равна 512 и более битам, то Виктор ничего не сможет узнать о закрытом ключе Пегги, кроме того, что Пегги действительно его знает.

Улучшения протокола

В протокол можно встроить идентификационные данные. Пусть I — это двоичная строка, представляющая идентификатор Пегги: имя, адрес, номер социального страхования, размер головного убора, любимый сорт прохладительного напитка и другая личная информация. Используем одностороннюю хеш-функцию $H(x)$ для вычисления $H(I, j)$, где j — небольшое число, добавленное к строке I. Найдем набор чисел j, для которых $H(I, j)$ представляет собой квадратичный вычет по модулю n. Эти значения $H(I, j)$ становятся строкой $v_1, v_2, \ldots, v_k$ (числа j не обязаны быть квадратичными вычетами). Теперь открытым ключом Пегги является строка I, и список чисел j. Пегги посылает строку I и список чисел j Виктору перед первым шагом протокола (в качестве альтернативы Виктор может загрузить эти значения с какой-нибудь открытой доски объявлений). Виктор генерирует строку $v_1, v_2, \ldots, v_k$ по значениям $H(I, j)$.

Теперь, после успешного завершения протокола, Виктор будет убежден, что Трент, которому известно разложение модуля на множители, сертифицировал связь между строкой I и Пегги, предоставив ей квадратные корни из v_i, полученные из строки I (см. раздел 5.2.) Фейге, Фиат и Шамир добавили следующие замечания [544, 545]:

Для неидеальных хеш-функций можно посоветовать рандомизировать строку I, добавляя к ней длинную случайную строку R. Эта строка выбирается арбитром и открывается Виктору вместе со строкой I.

В типичных реализациях число k должно быть от 1 до 18. Большие значения k могут уменьшить время и снизить трудности связи, уменьшая количество раундов.

Длина числа n должна быть не менее 512 битов. (Конечно, с тех пор в области факторизации достигнуты значительные успехи.)

Если каждый пользователь выберет свое собственное число n и опубликует его в файле открытых ключей, то можно обойтись и без арбитра. Однако такой RSA-подобный вариант делает схему гораздо менее удобной.

Схема подписи Фиата–Шамира

Превращение этой схемы идентификации в схему подписи — по существу, вопрос замены Виктора хеш-функцией. Главным преимуществом схемы цифровой подписи Фиата–Шамира над алгоритмом RSA является ее скорость:

для схемы Фиата–Шамира требуется всего лишь 1–4 процента модульных умножений, используемых в алгоритме RSA. В этом протоколе снова вернемся к Алисе и Бобу.

Используемые параметры не отличаются от параметров схемы идентификации. Выберем n — произведение двух больших простых чисел. Сгенерируем открытый ключ $v_1, v_2, ..., v_k$ и закрытый ключ $s_1, s_2, ..., s_k$, где $s_i = sqrt\left(v_i^{-1}\right) \bmod n$.

1. Алиса выбирает t случайных целых чисел $r_1, r_2, ..., r_t$ в диапазоне от 1 до n и вычисляет числа $x_1, x_2, ..., x_t$, такие, что $x_i = r_i^2 \bmod n$.
2. Алиса хеширует объединение сообщения и строки x_i, создавая поток битов: $H\left(m, x_1, x_2, ..., x_t\right)$. Она использует первые kt битов этой строки в качестве значений b_{ij}, где i изменяется от 1 до t, а j — от 1 до k.
3. Алиса вычисляет числа $y_1, y_2, ..., y_t$, где
$$y_i = r_i \left(s_1^{b_{i1}} s_2^{b_{i2}} ... s_k^{b_{ik}}\right) \bmod n .$$
(Для каждого i она перемножает значения s_i в зависимости от случайных значений b_{ij}. Если $b_{ij} = 1$, то s_i используется в вычислениях, если $b_{ij} = 0$, то нет.)
4. Алиса посылает Бобу число m, все биты b_{ij} и все значения y_i. У Боба уже есть открытый ключ Алисы, $v_1, v_2, ..., v_k$.
5. Боб вычисляет числа $z_1, z_2, ..., z_t$, где
$$z_i = y_i^2 \left(v_1^{b_{i1}} v_2^{b_{i2}} ... v_k^{b_{ik}}\right) \bmod n .$$
(Как и прежде, Боб выполняет умножение в зависимости от значений b_{ij}.) Также обратите внимание на то, что z_i должно быть равно x_i.
6. Боб проверяет, что первые kt битов $H\left(m, z_1, z_2 ..., z_t\right)$ — это значения b_{ij}, которые прислала ему Алиса.

Как и в схеме идентификации, безопасность схемы подписи пропорциональна $1/2^{kt}$. Она также зависит от сложности факторизации числа n. Фиат и Шамир показали, что подделка подписи упрощается, если сложность факторизации числа n значительно меньше 2^{kt}. Кроме того, из-за атаки на основе парадокса дней рождения (см. раздел 18.1) они рекомендуют повысить величину kt с 20 до (по крайней мере) 72, предлагая $k = 9$ и $t = 8$.

Улучшенная схема подписи Фиата–Шамира

Сильвия Микали (Silvia Micali) и Ади Шамир улучшили протокол Фиата–Шамира [1088]. Они выбирали числа $v_1, v_2, \ldots, v_k$ так, чтобы они были первыми k простыми числами:

$$v_1 = 2, v_2 = 3, v_3 = 5 \ldots$$

Это — открытый ключ. Закрытым ключом, $s_1, s_2, \ldots, s_k$, являются случайные квадратные корни, определяемые по формуле

$$s_i = sqrt\left(v_i^{-1}\right) \bmod n .$$

В этой версии у каждого участника должен быть свое число n. Такая модификация упрощает проверку подписей, не влияя на время генерации подписей и их стойкость.

Другие улучшения

На основе алгоритма Фиата–Шамира разработана N-сторонняя схема идентификации [264]. Два других улучшения схемы Фиата–Шамира описаны в [1218]. Еще один вариант изложен в [1368].

Схема идентификации Ота–Окамото

Этот протокол является вариантом схемы идентификации Фейге–Фиата–Шамира. Его стойкость основана на сложности факторизации целых чисел [1198, 1199]. Эти же авторы разработали схему с несколькими подписями (см. раздел 23.1), с помощью которой разные люди могут последовательно ставить цифровые подписи [1200]. Эта схема была предложена для реализации на интеллектуальных карточках [850].

Патенты

Схема Фиата–Шамира запатентована [1427]. Желающим получить лицензию на алгоритм необходимо обратиться по адресу Yeda Research and Development, The Weizmann Institute of Science, Rehovot 76100, Israel.

21.2. Схема Гиллу–Кискате

Схема Фейге–Фиата–Шамира была первым практическим протоколом идентификации. Этот протокол минимизировал вычисления, увеличивая количество итераций и аккредитаций на итерацию. Для ряда реализаций, например, для интеллектуальных карточек, это неприемлемо. Обмены информацией с внешним миром требуют времени, а хранение данных для каждой аккредитации может быстро исчерпать ограниченные возможности карточки.

Луи Гиллу (Louis Guillou) и Жан-Жак Кискате (Jean-Jacques Quisquater) разработали алгоритм идентификации с нулевым разглашением, который больше подходит для подобных приложений [670, 1280]. Обмены информацией между Пегги и Виктором, а также параллельные аккредитации в каждом обмене сведены к абсолютному минимуму: для каждого доказательства существует только один обмен, в котором предусмотрена только одна аккредитация. Для достижения того же уровня безопасности при использовании схемы Гиллу–Кискате потребуется выполнить в три раза больше вычислений, чем в схеме Фейге–Фиата–Шамира. Как и схему Фейге–Фиата–Шамира, этот алгоритм идентификации можно превратить в алгоритм цифровой подписи.

Схема идентификации Гиллу–Кискате

Предположим, что роль Пегги играет интеллектуальная карточка, которая собирается доказать свою подлинность Виктору. Идентификация Пегги проводится по ряду атрибутов, представляющих собой строку данных, содержащих название карточки, период ее действия, номер банковского счета и другие данные, подтверждающие ее правомочность. Эта битовая строка называется J. (В реальности строка атрибутов может быть очень длинной, а в качестве строки J используется ее хеш-значение. Такое усложнение никак не влияет на протокол.) Эта строка аналогична открытому ключу. Другой открытой информацией, общей для всех "Пегги", которые могут использовать это приложение, является показатель степени v и модуль n, где n — произведение двух секретных простых чисел. Закрытым ключом служит число B, удовлетворяющее условию $JB^v \equiv 1 \pmod{n}$.

Пегги посылает Виктору свои атрибуты J. Теперь она хочет доказать Виктору, что это именно ее атрибуты. Для этого она должна убедить Виктора, что ей известно число B. Вот как выглядит этот протокол.

1. Пегги выбирает случайное целое r, лежащее в диапазоне от 1 до $n-1$. Она вычисляет число $T = r^v \bmod n$ и отправляет его Виктору.
2. Виктор выбирает случайное целое d, находящееся в диапазоне от 0 до $v-1$, и посылает число d Пегги.
3. Пегги вычисляет $D = rB^d \bmod n$ и посылает его Виктору.
4. Виктор вычисляет число $T' = D^v J^d \bmod n$. Если $T \equiv T' \pmod{n}$, то подлинность Пегги доказана.

Математическое доказательство этого протокола не слишком сложное:

$$T' = D^v J^d = \left(rB^d\right)^v J^d = r^v B^{dv} J^d = r^v \left(B^v J\right)^d = r^v = r' \equiv T \pmod{n},$$

так как $JB^v \equiv 1 \pmod{n}$.

Схема подписи Гиллу–Кискате

Эту схему идентификации можно преобразовать в схему подписи, также пригодную для реализации в интеллектуальных карточках [671, 672]. Открытый и закрытый ключи не меняются. Вот как выглядит протокол.

1. Алиса выбирает случайное целое r, находящееся в диапазоне от 1 до $n-1$. Она вычисляет $T = r^v \bmod n$.
2. Алиса вычисляет число $d = H(M, T)$, где M — подписываемое сообщение, а $H(x)$ — односторонняя хеш-функция. Значение d, полученное с помощью хеш-функции, должно лежать в диапазоне от 0 до $v-1$ [1280]. Если значение хеш-функции выходит за пределы этого диапазона, то оно должно быть приведено по модулю v.
3. Алиса вычисляет число $D = rB^d \bmod n$. Подпись состоит из сообщения M, двух вычисленных значений, d и D, и ее атрибутов J. Алиса посылает подпись Бобу.
4. Боб вычисляет $T' = D^v J^d \bmod n$. Затем он вычисляет $d' = H\left(M, T'\right)$. Если $d = d'$, то Алиса знает число B и ее подпись является действительной.

Несколько подписей

Что произойдет, если несколько человек захотят подписать один и тот же документ? Проще всего, чтобы они подписали его по-отдельности, но рассматриваемая схема подписи делает это лучше. Пусть Алиса и Боб подписывают документ, а Кэрол проверяет подписи, но в процесс подписания может быть вовлечено произвольное количество людей. Как и раньше, Алиса и Боб обладают уникальными значениями J и B: (J_A, B_A) и (J_B, B_B). Значения n и v являются общими для всей системы.

1. Алиса выбирает случайное целое r_A, находящееся в диапазоне от 1 до $n-1$. Она вычисляет число $T_A = r_A^v \bmod n$ и посылает число T_A Бобу.
2. Боб выбирает случайное целое r_B, находящееся в диапазоне от 1 до $n-1$. Он вычисляет число $T_B = r_B^v \bmod n$ и посылает число T_B Алисе.
3. Алиса и Боб вычисляют число $T = \left(T_A \times T_B\right) \bmod n$.
4. И Алиса, и Боб вычисляют $d = H(M, T)$, где M — подписываемое сообщение, а $H(x)$ — односторонняя хеш-функция. Значение d, полученное с помощью хеш-функции, должно лежать в диапазоне от 0 до $v-1$ [1280]. Если значение хеш-функции выходит за пределы этого диапазона, то он должен быть приведен по модулю v.
5. Алиса вычисляет $D_A = r_A B_A^d \bmod n$ и посылает D_A Бобу.

6. Боб вычисляет $D_B = r_B B_B^d \bmod n$ и посылает D_B Алисе.
7. И Алиса, и Боб вычисляют $D = D_A D_B \bmod n$. Подпись состоит из сообщения M, двух вычисленных значений, d и D, и атрибутов обоих подписывающих: J_A и J_B.
8. Кэрол вычисляет $J = J_A J_B \bmod n$.
9. Кэрол вычисляет $T' = D^v J^d \bmod n$. Затем она вычисляет $d' = H(M, T')$. Если $d \equiv d'$, то множественная подпись является действительной.

Этот протокол может быть расширен на произвольное количество людей. Для этого люди, подписывающие сообщение, должны перемножить свои значения T_i на шаге 3 и свои значения D_i на шаге 7. Для того чтобы проверить множественную подпись, необходимо на шаге 8 перемножить значения всех подписывающих J_i. Либо все подписи правильны, либо существует по крайней мере одна неправильная подпись.

21.3. Схема Шнорра

Стойкость схемы проверки подлинности и подписи, разработанная Клаусом Шнорром [1396,1397], опирается на трудность вычисления дискретных логарифмов. Для генерации пары ключей сначала выбираются два простых числа, p и q, так, чтобы число q было множителем числа $p-1$. Затем выбирается число a, не равное единице, такое, что $a^q \equiv 1 \pmod p$. Все эти числа можно свободно опубликовать и использовать в группе пользователей.

Для генерации конкретной пары ключей выбирается случайное число, меньшее q. Оно является закрытым ключом, s. Затем вычисляется открытый ключ $v = a^{-s} \bmod p$.

Протокол проверки подлинности

1. Пегги выбирает случайное число r, меньшее q, и вычисляет $x = a^r \bmod p$. Эти вычисления являются предварительными и могут быть выполнены задолго до появления Виктора.
2. Пегги посылает Виктору число x.
3. Виктор посылает Пегги случайное число e из диапазона $0 - 2^t - 1$. (Что такое t, я объясню позднее.)
4. Пегги вычисляет число $y = (r + se) \bmod q$ и посылает его Виктору.
5. Виктор проверяет условие $x = a^y v^e \bmod p$.

Стойкость алгоритма зависит от параметра t. Сложность взлома алгоритма равна примерно 2^t. Шнорр советует использовать число p длиной около 512 битов, q — около 140 битов и t — 72 бита.

Протокол цифровой подписи

Алгоритм Шнорра можно использовать и в качестве протокола цифровой подписи сообщения M. В этом случае используется та же самая пара ключей, но добавляется одностороняя хеш-функция $H(M)$.

1. Алиса выбирает случайное число r, меньшее q, и вычисляет $x = a^r \bmod p$. Это стадия предварительных вычислений.
2. Алиса объединяет M и x и хеширует результат:
$$e = H(M, x).$$
3. Алиса вычисляет число $y = (r + se) \bmod q$. Подписью являются значения e и y, которые она посылает Бобу.
4. Боб вычисляет число $x' = a^y v^e \bmod p$. Затем он проверяет, что хеш-значение для конкатенации M и x' равно e:
$$e = H\left(M, x'\right).$$
5. Если это так, то он считает подпись верной.

В своей работе Шнорр приводит следующие новые свойства своего алгоритма:

> Большая часть вычислений, необходимых для генерации подписи и не зависящих от подписываемого сообщения, может быть выполнена на стадии предварительных вычислений. Следовательно, эти вычисления можно выполнить во время простоя и они не влияют на скорость подписания. Атака, направленная на стадию предварительных вычислений, рассматривается в [475], но я не думаю, что она имеет практическую ценность.
>
> При одинаковом уровне безопасности длина подписей в схеме Шнорра короче, чем в алгоритме RSA. Например, при 140-битовом числе q длина подписей равна всего лишь 212 битам, т.е. меньше половины длины подписей RSA. Подписи по схеме Шнорра также намного короче подписей по схеме Эль-Гамаля.

Конечно, из практических соображений количество битов, используемых в этой схеме, может быть уменьшено: например, для схемы идентификации, в которой мошенник должен выполнить атаку в онлайновом режиме всего за несколько секунд (сравните со схемой подписи, когда мошенник может годами вести расчеты, чтобы выполнить подлог).

Модификация, выполненная Эрни Брикеллом (Ernie Brickell) и Кевином МакКерли (Kevin McCurley), повысила стойкость этого алгоритма [265].

Патенты

Схема Шнорра запатентована в США [1398] и во многих других странах. В 1993 г. компания PKP приобрела общемировые права на этот патент (см. раздел 25.5). Срок действия патента США истекает 19 февраля 2008 г.

21.4. Преобразование схем идентификации в схемы подписи

Стандартный метод преобразования схемы идентификации в схему подписи — замена Виктора односторонней хеш-функцией. Перед подписанием сообщение не хешируется, вместо этого хеширование встраивается в алгоритм подписи. В принципе, такую манипуляцию можно проделать с любой схемой идентификации.