Лекция №5

Часть 1. Схема криптографической подписи

Елена Киршанова

Курс “Основы криптографии”

Агенда

- Зачет в субботу 12/11 в 10:10 в Webinar
  `https://events.webinar.ru/58831339/1604249105`
- Доп. защита лабраторных в пятницу 11/11 в 10:00 в Webinar
  `https://events.webinar.ru/58831339/2144800577`

Схема цифровой подписи: зачем нужна?

Обмен ключами + Симметрическое шифрование = конфидециальность

Но

- протокол обмена ключами Диффи-Хэллмана подвержен активным атакам
- он не обеспечивает целостность передаваемых данных
- он не обеспечивает аутентификацию

Наша цель: обеспечить целостность и аутентификацию, как схема MAC, но в асимметриченом открытым мире.

Асимметричная версия MAC'а – криптогрфическая схема подписи

Схема подписи: определение

Схема подписи состоит из трех ppt алгоритмов

- Генерация ключа: $(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$
  vk – ключ верификации (открытый), sk – подписывающий ключ (секретный)
- Генерация подписи: $\sigma \leftarrow \mathsf{Sign}(m, \mathsf{sk})$
- Верификация: $\mathsf{Ver}(m, \sigma, \mathsf{vk})$ выдает $\{\mathsf{accept}, \mathsf{reject}\}$.

Здесь, $m \in \mathcal{M}$ – сообщение, которое должно быть подписано

Корректность: $\forall m, \forall(\mathsf{sk}, \mathsf{vk}) \leftarrow \mathsf{KeyGen}()$:

$$\mathsf{Ver}(m, \mathsf{Sign}(m, \mathsf{sk}), \mathsf{vk}) = \mathsf{accept}$$

Схема подписи: безопасность

2 типа атак на выбранное сообщение:

I. Экзистенциальная подделка (Existential forgery)

- атакующий может запросить подписать любое сообщение m
- ppt атакующий не должен сформировать корректную пару (m, σ) для нового m, т.е., для m, для которого он не запрашивал подпись

Схема подписи: безопасность

2 типа атак на выбранное сообщение:

I. Экзистенциальная подделка (Existential forgery)

- атакующий может запросить подписать любое сообщение m
- ppt атакующий не должен сформировать корректную пару (m, σ) для нового m, т.е., для m, для которого он не запрашивал подпись

II. Сильная экзистенциальная подделка (Strong Existential forgery)

- атакующий может запросить подписать любое сообщение m
- ppt атакующий не должен сформировать корректную пару (m, σ) даже для сообщений, на которые была запрошена подпись, т.е., (m, σ') является корректной атакой, даже если злоумышленник знает (m, σ).

Из цифровой подписи, безопасной в первой (слабой) модели, можно сделать схему подписи, безопасную во второй, более сильной модели

Безопасность UF-CMA (Unforgeability under Chosen Message Attack)

$$\Pi = (\mathsf{KeyGen}, \mathsf{Sign}, \mathsf{Ver}) \text{ – схема подписи}$$

Челленджер $\mathcal{C}$		Атакующий $\mathcal{A}$
$(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\lambda)$		$\mathcal{M} \leftarrow \emptyset$
	$\xleftarrow{m}$	
$\sigma = \mathsf{Sign}(m, \mathsf{sk})$	$\xrightarrow{\sigma}$	$\mathcal{M} \leftarrow \mathcal{M} \cup \{m\}$
	$\xleftarrow{(m', \sigma')}$	$m' \notin \mathcal{M}$

$\mathtt{W}_{\Pi,\mathcal{A}}$ – событие $\mathsf{Ver}(\mathsf{vk}, \sigma') = \mathsf{accept}$.

$\mathtt{SigAdv} = \Pr[\mathtt{W}_{\Pi,\mathcal{A}}]$ – выигрыш $\mathcal{A}$.

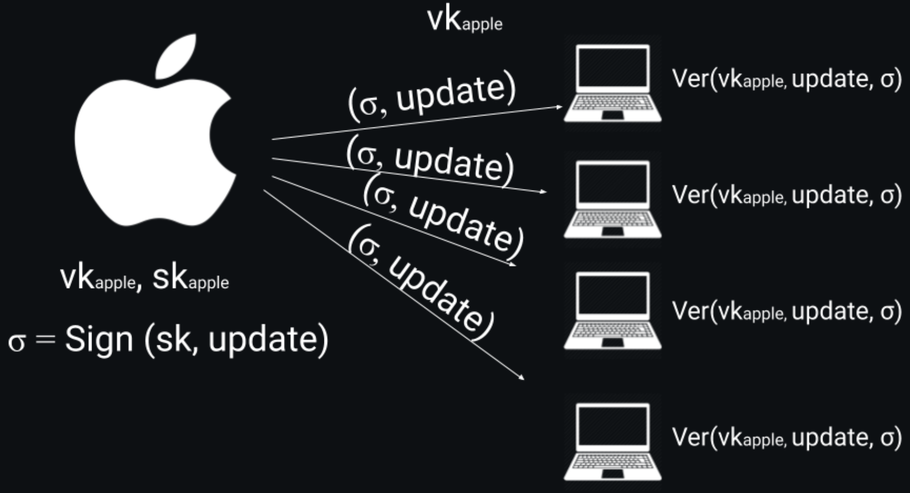
Схема подписи Π безопасна в модели UF-CMA, если $\forall$ ppt $\mathcal{A}$:

$$\mathtt{SigAdv} = \mathrm{negl}(\lambda).$$

Еще о безопасности

- Неотказ от авторства (Non-repudiation)
  – Подписывающий “привязан” к своим подписям.
- Стойкость относительно дубликатов подписывающих ключей (Duplicate Signature Key Selection,DSKS)
  – Атакующий, имея (m, σ), может сгенерировать пару$(\mathsf{vk}', \mathsf{sk}')$, т.ч. (m, σ) – корректная пара относительно $(\mathsf{vk}', \mathsf{sk}')$.
  – Для предотвращения такой атаки подписывающий конкатинирует сообщение со своим открытым ключом

Применение цифровой подписи: обновление софта



Схемы подписи на практике

1. RSA
   – основана на сложности задачи факторизации
   – быстрая процедура Ver, медленные Sign, KeyGen; длинный ключи, подписи

2. (EC)DSA= (Elliptic Curve) Digital Signature Algorithm
   – основана на сложности задачи dlog
   – медленнее Ver, быстрее Sign, KeyGen
   – в ECDSA меньшего размера ключи и подписи

Схемы подписи на практике

1. RSA
   – основана на сложности задачи факторизации
   – быстрая процедура Ver, медленные Sign, KeyGen; длинный ключи, подписи

2. (EC)DSA= (Elliptic Curve) Digital Signature Algorithm
   – основана на сложности задачи dlog
   – медленнее Ver, быстрее Sign, KeyGen
   – в ECDSA меньшего размера ключи и подписи

3. ГОСТ Р 34.10-2012
   – аналогичен ECDSA
   – старый ГОСТ Р 34.10-94 аналогичен DSA

Схемы подписи на практике

1. RSA
   – основана на сложности задачи факторизации
   – быстрая процедура Ver, медленные Sign, KeyGen; длинный ключи, подписи
2. (EC)DSA= (Elliptic Curve) Digital Signature Algorithm
   – основана на сложности задачи dlog
   – медленнее Ver, быстрее Sign, KeyGen
   – в ECDSA меньшего размера ключи и подписи
3. ГОСТ Р 34.10-2012
   – аналогичен ECDSA
   – старый ГОСТ Р 34.10-94 аналогичен DSA

Размеры ключей (в битах):

Security lvl.	ECDSA / ГОСТ'12	RSA/DSA
80	160	1024
128	256	4096
256	512	15360

Часть II

Арифметика в кольце целых чисел

Арифметика в кольце целых

Положим $N = p \cdot q$, где p, q – большие простые числа

- $\mathbb{Z}_N = \{0, 1, \ldots, N-1\}$ – кольцо
- Элементы $\mathbb{Z}_N$ складываются и умножаются по модулю N.
  Пример: $N = 15$

$$11 + 6 \bmod N = \text{rem}(17, 15) = 2$$
$$6 \cdot 7 \bmod N = \text{rem}(42, 15) = 12$$

- Не для всякого ненулевого $x \in \mathbb{Z}_N$ существует обратимый!
  Множество обратимых элементов: $\mathbb{Z}_N^* = \{x \in \mathbb{Z}_N \mid \text{НОД}(x, N) = 1\}$.
  Пример: $3, 6, 9, 5, 10, 12 \notin \mathbb{Z}_N^*$.
  $\mathbb{Z}_N^* = \{1, 2, 4, 7, 8, 11, 13, 14\}$.

Структура $\mathbb{Z}_N^*$

- $\phi(N) = |\mathbb{Z}_N^*|$ – функция Эйлера
  - N – простое, $\phi(N) = N - 1$
  - $N = p_1^{e_1} \cdot p_n^{e_n}$, $\phi(N) = N \cdot \prod_i \left(1 - \frac{1}{p_i}\right)$.
  - для $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$.

  Пример: $|\mathbb{Z}_N^*| = |\{1, 2, 4, 7, 8, 11, 13, 14\}| = 2 \cdot 4 = 8$.
- Теорема Эйлера: для всех $a \in \mathbb{Z}_N^*$

$$a^{\phi(n)} = 1 \bmod N$$

Теорема Ферма: $a^{p-1} = 1 \bmod p$ для простого p.

Простые и трудные задачи в $\mathbb{Z}_N$

$N = p \cdot q$, p, q – по $\approx$ 1024/2048 бит каждое.

В $\mathbb{Z}_N$ следующие операции эффективные

– сложение, умножение, нахождение обратного (если существует, проверить несуществование)

– возведение в степерь $g^r \bmod N$

Сегодня считаются трудными задачи

– факторизация: нахождение p, q

– вычисление квадратного корня в $\mathbb{Z}_N$ (эквивалентно факторизации)

– вычисление e-го корня в $\mathbb{Z}_N$ при $\gcd(e, \phi(N)) = 1$

Часть III

Подпись RSA

Генерация ключей в подписи RSA

Возьмём $\ell > 2$–целое и $e > 2$ – нечетное целое (на практике $e = 65537$)

RSAGen(ℓ, e) :

1. Сгенерировать ℓ-битное целое p т.ч. $\gcd(p-1, e) = 1$
2. Сгенерировать ℓ-битное цело $q \neq p$ т.ч. $\gcd(q-1, e) = 1$
3. $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$
4. $d = e^{-1} \bmod \phi(N)$
5. Вывод: $\mathsf{vk} = (N, e), \mathsf{sk} = (N, d)$

Генерация ключей в подписи RSA

Возьмём $\ell > 2$–целое и $e > 2$ – нечетное целое (на практике $e = 65537$)

RSAGen(ℓ, e) :

1. Сгенерировать ℓ-битное целое p т.ч. $\gcd(p-1, e) = 1$
2. Сгенерировать ℓ-битное цело $q \neq p$ т.ч. $\gcd(q-1, e) = 1$
3. $N = p \cdot q$, $\phi(N) = (p-1)(q-1)$
4. $d = e^{-1} \mod \phi(N)$
5. Вывод: $\mathsf{vk} = (N, e), \mathsf{sk} = (N, d)$

- $\exists$ ppt алгоритм генерации простых чисел ($\exists$ ppt алгоритмы теста на простоту)
- Шаг 4 корректен: $d \in \mathbb{Z}_N^*$, т.к.

$$\gcd(p-1, e) = \gcd(q-1, e) = 1 \implies \gcd((p-1)(q-1), e) = 1.$$

- на p, q должны удовлетворять большому числу условий

Не пытайтесь повторить RSAGen самостоятельно.

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$ – криптографическая хэш-функция

I. $\mathsf{RSASign}(\mathsf{sk} = (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$– криптографическая хэш-функция

I. $\mathsf{RSASign}(\mathsf{sk} = (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. $\mathsf{RSAVerify}(\mathsf{vk} = (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\texttt{return}(y' == \mathcal{H}(m))$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$ – криптографическая хэш-функция

I. $\mathsf{RSASign}(\mathsf{sk} = (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. $\mathsf{RSAVerify}(\mathsf{vk} = (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\mathtt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$ – криптографическая хэш-функция

I. $\mathsf{RSASign}(\mathsf{sk} = (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. $\mathsf{RSAVerify}(\mathsf{vk} = (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\texttt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

Док-во: для $k \in \mathbb{Z}$

$$ed = 1 + k\phi(N) = 1 + k(p-1)(q-1)$$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$– криптографическая хэш-функция

I. RSASign(sk $= (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. RSAVerify(vk $= (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\mathtt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

Док-во: для $k \in \mathbb{Z}$

$$ed = 1 + k\phi(N) = 1 + k(p-1)(q-1)$$

$$x^{p-1} = 1 \bmod p \quad \text{(Ferma't thm.)}$$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$– криптографическая хэш-функция

I. RSASign(sk $= (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. RSAVerify(vk $= (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\texttt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

Док-во: для $k \in \mathbb{Z}$

$$ed = 1 + k\phi(N) = 1 + k(p-1)(q-1)$$

$$x^{p-1} = 1 \bmod p \quad \text{(Ferma't thm.)}$$

$$x^{ed} = x^{1+k(p-1)(q-1)} =$$

$$x \cdot (x^{p-1})^{q-1} = x \bmod p$$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$– криптографическая хэш-функция

I. $\text{RSASign}(\text{sk} = (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. $\text{RSAVerify}(\text{vk} = (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\texttt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

Док-во: для $k \in \mathbb{Z}$

$$ed = 1 + k\phi(N) = 1 + k(p-1)(q-1)$$

$x^{p-1} = 1 \bmod p$ (Ferma't thm.)

$$x^{ed} = x^{1+k(p-1)(q-1)} =$$

$$x \cdot (x^{p-1})^{q-1} = x \bmod p$$

Аналог., $x^{ed} = x \bmod q$

RSA Signature Generation and Verification

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_N^*$ – криптографическая хэш-функция

I. RSASign(sk $= (N, d), m)$:

1. $y = \mathcal{H}(m) \in \mathbb{Z}_N^*$
2. $\sigma = y^d \bmod N$

II. RSAVerify(vk $= (N, e), m, \sigma)$:

1. $y' = \sigma^e \bmod N$
2. $\texttt{return}(y' == \mathcal{H}(m))$

Корректность: Для $N = pq$ и e, d т.ч. $ed = 1 \bmod \phi(N)$ и для всех $x \in \mathbb{Z}$

$$x^{ed} = x \bmod N$$

Док-во: для $k \in \mathbb{Z}$

$$ed = 1 + k\phi(N) = 1 + k(p-1)(q-1)$$

$$x^{p-1} = 1 \bmod p \quad \text{(Ferma't thm.)}$$

$$x^{ed} = x^{1+k(p-1)(q-1)} =$$

$$x \cdot (x^{p-1})^{q-1} = x \bmod p$$

Аналог., $x^{ed} = x \bmod q$

$$\implies p, q \,|\, x^{ed} - x$$

$$\implies x^{ed} = x \bmod p \cdot q$$

Без $\mathcal{H}$ схема тривиально взламывается!

Безопасность подписи RSA

RSA предположение сложности: Не существует ppt алгоритма, который, получив на вход (N, m, m^e) для случайного $m \in \mathbb{Z}_N^*$, выдаёт m.

Факторизация $N \implies$ вычисление e-го корня.

Обратная редукция не доказана!

Самый быстрый сегодня алгоритм факторизации – General Number Field sievie – со сложностью

$$\sim e^{(\lg N)^{1/3}}$$

Теорема: Схема подписи (RSAGen, RSASign, RSAVerify) безопасна в модели UF-CMA, если предположение RSA корректно и $\mathcal{H}$ является случайным оракулом.

Стандарты

Вариант RSA подписи станартизованы под именем PKCS (Public Key Cryptography Standards)
`https://en.wikipedia.org/wiki/PKCS`

Версия 2.2 (крайняя) PKCS #1 включает

- RSASSA-PSS
  SSA = Signature Scheme with Appendix
  PSS = Probabilistic Signature Scheme
- RSASSA-PKCS1-v1_5 (существуют атаки)

Часть IV

Слепая подпись. Сертификаты

Слепая подпись RSA

Сценарий: А необходима подпись банка (Б) на осуществлении транзакции. При этом Б не должен знать о том, что именно он подписывает. Б осуществляет слепую подпись.

$\mathcal{H} : \{0,1\}^* \to \mathbb{Z}_p^*$ – криптографическая хэш-функция
m – транзакция

А		Б
		$\mathsf{sk} = (N, d), \mathsf{vk} = (N, e) \leftarrow \texttt{RSA.KeyGen}()$
$r \xleftarrow{\$} \mathbb{Z}_N$		
$m' = \mathcal{H}(m) \cdot r^e$	$\xrightarrow{m'}$	$\sigma' = (m')^d$
$\sigma = \sigma'/r$	$\xleftarrow{\sigma'}$	

- А получает корректную подпись для m'
- Значение r^e маскирует исходное сообщение (Б не знает ничего о m)

Слепая подпись

- Слепая подпись безопасна, если ppt атакующий, зная vk и Q слепых подписей, не может сгенерировать $Q+1$ пару (m, σ)
- Безопасность слепой подписи RSA основана на задача 1MRSA:
  - A получает значения $w = \hat{v}^{1/e}$ от Челленджера для $\hat{v}$ по своему выбору
  - Задача A: вычислить $v^{1/e}$ для какого-либо одного $v \neq \hat{v}$, выбранного Челленджером
- Слепая подпись может быть построена на основе подписи Шнорра
- Используются в протоколах Anonymous Credentials, E-cash.

Применение криптографических подписей

Подписи на практике

1. RSA
   длинные ключи и подписи; быстрая верификация
2. ECDSA, ГОСТ
   короткие ключи и подписи; верификация медленнее

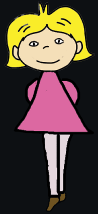
RSA хороша для Сертификатов, ECDSA/ГОСТ – для имейлов.

Сертификаты

Задача сертификата: связать публичный ключ с человеком/машиной.

Алиса

$\mathsf{pk}_{\mathtt{A}}$



Сертификационный центр (CA)

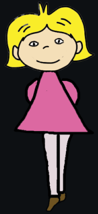
$\mathsf{vk}_{\mathtt{CA}}$, $\mathsf{sk}_{\mathtt{CA}}$

Сертификаты

Задача сертификата: связать публичный ключ с человеком/машиной.

Алиса

$\mathsf{pk}_{\mathtt{A}}$



Сертификационный центр (CA)

$\mathsf{vk}_{\mathtt{CA}}, \mathsf{sk}_{\mathtt{CA}}$

$$\xrightarrow{\text{Certificate Signing Request}}$$

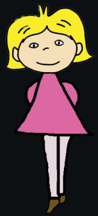
$\mathtt{id},\ \mathtt{email},\ \mathsf{pk}_{\mathtt{A}}$

Сертификаты

Задача сертификата: связать публичный ключ с человеком/машиной.

Алиса

$\mathsf{pk}_{\mathtt{A}}$



Сертификационный центр (CA)

$\mathsf{vk}_{\mathtt{CA}}$, $\mathsf{sk}_{\mathtt{CA}}$

Certificate Signing Request $\longrightarrow$

`id, email,` $\mathsf{pk}_{\mathtt{A}}$

Сертификация id Алисы:

Создание `cert`

Подпись `cert`

$\longleftarrow$ $\mathtt{cert}, \sigma{=}\mathsf{Sign}(\mathsf{sk}_{\mathtt{CA}},\mathtt{cert})$

Для осуществления аутентифицированного канала с Алисой верифицируется `cert`: $\mathsf{Ver}(\mathsf{vk}_{\mathtt{CA}}\,\mathtt{cert}, \sigma)$.

Если $\mathsf{Ver}(\mathsf{vk}_{\mathtt{CA}}\,\mathtt{cert}, \sigma) = \mathsf{accept}$, то $\mathsf{pk}_{\mathtt{A}}$ используется для коммуникации с Алисой.

Пример: сертификат X.509 certificate

Цепочки сертификатов

Корневой CA $\longrightarrow$ Промежуточный CA

сертифицирует промежуточные CA — сертифицирует клиентов

Сегодня мы имеем тысячи промежуточный CAs

Для противостояния CAs: certificate pinning:

1. Каждый браузер (клиент) поддерживает базу данных (pinning database): $(\mathtt{domain}, \mathtt{hash}_0, \mathtt{hash}_1, \ldots)$
2. Данные для БД предоставляет домен
3. Когда браузер стучится в домен, домен высылает ему цепочку своих сертификатов $\mathtt{cert}_0, \mathtt{cert}_1, \ldots$
4. Браузер вычисляет $\mathcal{H}(\mathtt{cert}_i)$ и верифицирует $\mathtt{hash}_i$.

Пример цепочки сертификатов