Лекция 8. Протоколы идентификации с нулевым разглашением (zero-knowledge idetification schemes)

Косолапов Ю.В.

ЮФУ

21 октября 2020 г.

Содержание

Проблемы протоколов типа «запрос-ответ»

Проблемы протоколов типа «запрос-ответ»

- **Высокие требования безопасности**: к криптографическим алгоритмам предъявляются высокие требования стойкости.

Проблемы протоколов типа «запрос-ответ»

- **Высокие требования безопасности**: к криптографическим алгоритмам предъявляются высокие требования стойкости.
- **Сложные вычисления**: для вычисления ответа доказывающая сторона должна выполнить сложные (ресурсоемкие) вычисления.

Проблемы протоколов типа «запрос-ответ»

- **Высокие требования безопасности**: к криптографическим алгоритмам предъявляются высокие требования стойкости.
- **Сложные вычисления**: для вычисления ответа доказывающая сторона должна выполнить сложные (ресурсоемкие) вычисления.

Example

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи

$$
\begin{aligned}
P &\to V : \textsc{id}(P) \\
V &\to P : \textsc{nonce} \\
P &\to V : SIGN_{sk_P}(\textsc{nonce}).
\end{aligned}
$$

Проблемы протоколов типа «запрос-ответ»

- **Высокие требования безопасности**: к криптографическим алгоритмам предъявляются высокие требования стойкости.
- **Сложные вычисления**: для вычисления ответа доказывающая сторона должна выполнить сложные (ресурсоемкие) вычисления.

Example

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи

$$\begin{aligned} P &\to V : \textsc{id}(P) \\ V &\to P : \textsc{nonce} \\ P &\to V : SIGN_{sk_P}(\textsc{nonce}). \end{aligned}$$

- **Высокие требования безопасности**: стойкость к атакам по подобранному открытому тексту (невозможно узнать ключ подписи sk_P или подготовить пару (m, σ) так, чтобы σ была верной подписью для m).

Проблемы протоколов типа «запрос-ответ»

- **Высокие требования безопасности**: к криптографическим алгоритмам предъявляются высокие требования стойкости.
- **Сложные вычисления**: для вычисления ответа доказывающая сторона должна выполнить сложные (ресурсоемкие) вычисления.

Example

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи

$$P \to V : \textsc{id}(P)$$
$$V \to P : \textsc{nonce}$$
$$P \to V : SIGN_{sk_P}(\textsc{nonce}).$$

- **Высокие требования безопасности**: стойкость к атакам по подобранному открытому тексту (невозможно узнать ключ подписи sk_P или подготовить пару (m, σ) так, чтобы σ была верной подписью для m).
- **Сложные вычисления**: современные схемы подписи основаны на ресурсоемкой арифметике (работа в полях/кольцах/группах очень большой мощности).

Идея ZK-протоколов идентификации (ZK – zero knowledge)

Предположения:

Идея ZK-протоколов идентификации (ZK – zero knowledge)

Предположения:

- **Доказывающая сторона** P – **честная**: не отклоняется от протокола.

Идея ZK-протоколов идентификации (ZK – zero knowledge)

Предположения:

- **Доказывающая сторона** P – **честная**: не отклоняется от протокола.
- **Проверяющая сторона** V **может мошенничать**: может отклоняться от протокола, пытаясь узнать секрет доказывающей стороны (здесь **секрет** – это ключ, с помощью которого P, доказывает свою подлинность/подтверждает свой идентификатор).

Идея ZK-протоколов идентификации (ZK – zero knowledge)

Предположения:

- **Доказывающая сторона** P – **честная**: не отклоняется от протокола.
- **Проверяющая сторона** V **может мошенничать**: может отклоняться от протокола, пытаясь узнать секрет доказывающей стороны (здесь **секрет** – это ключ, с помощью которого P, доказывает свою подлинность/подтверждает свой идентификатор).

Цель: Проверяющая сторона не должна узнать из ответов доказывающей стороны **ничего полезного** для получения информации секрете доказывающей стороны.

Идея ZK-протоколов идентификации (ZK – zero knowledge)

Предположения:

- **Доказывающая сторона** P – **честная**: не отклоняется от протокола.
- **Проверяющая сторона** V **может мошенничать**: может отклоняться от протокола, пытаясь узнать секрет доказывающей стороны (здесь **секрет** – это ключ, с помощью которого P, доказывает свою подлинность/подтверждает свой идентификатор).

Цель: Проверяющая сторона не должна узнать из ответов доказывающей стороны **ничего полезного** для получения информации секрете доказывающей стороны.

ZK

Если ответы доказывающей стороны могут быть симулированы проверяющей стороной без вовлечения доказывающей стороны, то говорят, что ответы не несут какой-либо полезной информации. В этом случае проверяющая сторона получает **ноль знаний** из ответов доказывающей стороны (zero-knowledge).

Пример протокола, который не ZK

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи, sk_P – секретный ключ P, sk_P – публичный ключ P.

$$P \to V : \textsc{id}(P)$$
$$V \to P : \textsc{nonce}$$
$$P \to V : SIGN_{sk_P}(\textsc{nonce}).$$

Пример протокола, который не ZK

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи, sk_P – секретный ключ P, sk_P – публичный ключ P.

$$P \to V : \textsc{id}(P)$$
$$V \to P : \textsc{nonce}$$
$$P \to V : SIGN_{sk_P}(\textsc{nonce}).$$

- Проверяющая сторона может мошенничать: выбирать NONCE не случайно, а подбирать специальным образом (например, посылать значения 0,1,2,...),

Пример протокола, который не ZK

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи, sk_P – секретный ключ P, sk_P – публичный ключ P.

$$P \to V : \text{ID}(P)$$
$$V \to P : \text{NONCE}$$
$$P \to V : SIGN_{sk_P}(\text{NONCE}).$$

- Проверяющая сторона может мошенничать: выбирать NONCE не случайно, а подбирать специальным образом (например, посылать значения 0,1,2,...),
- Проверяющая сторона не может без участия P сгенерировать правильную подпись для любого текста (то есть **не может симулировать ответ** доказывающей стороны). Если бы V могла бы генерировать правильную подпись, то это равносильно тому, что она знает секретный ключ sk_P.

Пример протокола, который не ZK

Схема аутентификации типа «запрос-ответ» на основе цифровой подписи, sk_P – секретный ключ P, sk_P – публичный ключ P.

$$P \to V : \textsc{id}(P)$$
$$V \to P : \textsc{nonce}$$
$$P \to V : SIGN_{sk_P}(\textsc{nonce}).$$

- Проверяющая сторона может мошенничать: выбирать $\textsc{nonce}$ не случайно, а подбирать специальным образом (например, посылать значения 0,1,2,...),
- Проверяющая сторона не может без участия P сгенерировать правильную подпись для любого текста (то есть **не может симулировать ответ** доказывающей стороны). Если бы V могла бы генерировать правильную подпись, то это равносильно тому, что она знает секретный ключ sk_P.
- Поэтому пары

$$(\textsc{nonce}_1, SIGN_{sk_P}(\textsc{nonce}_1)), (\textsc{nonce}_2, SIGN_{sk_P}(\textsc{nonce}_2)), \dots$$

несут в себе какую-то **ненулевую** информацию о ключе (как извлечь эту информацию – это другой вопрос!). Поэтому этот протокол не является ZK-протоколом.

Протокол Шнорра (Schnorr)

Протокол Шнорра (Schnorr)

- Пусть $\langle g\rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g\rangle|$ – порядок этой группы (большое, желательно простое число).

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n$, $a = g^u$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n, \ a = g^u$
  2. $P \to V : a$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n$, $a = g^u$
  2. $P \to V : a$
  3. $V : c \in_R \{0, 1\}$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$
  6. $P \to V : r$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$
  6. $P \to V : r$
  7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah, c = 1 \end{cases}$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n$, $a = g^u$
  2. $P \to V : a$ — анонс
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$ — запрос
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$
  6. $P \to V : r$ — ответ
  7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah, c = 1 \end{cases}$

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P \to V : a$ — анонс
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$ — запрос
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$
  6. $P \to V : r$ — ответ
  7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah, c = 1 \end{cases}$

  Тройка $(a, c, r)(\in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n)$ называется *разговором* (talk).

Протокол Шнорра (Schnorr)

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
- **Одна итерация протокола Шнорра**:
  1. $P : u \in_R \mathbb{Z}_n$, $a = g^u$
  2. $P \to V : a$ — вручение обязательства (прячется ключ u)
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P : c$
  5. $P : r = \begin{cases} u, c = 0 \\ u + x, c = 1 \end{cases}$
  6. $P \to V : r$ — «раскрытие обязательства»
  7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah, c = 1 \end{cases}$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P' \to V : a$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P' : c$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P' : c$
  5. $P' : r = \begin{cases} u, c = 0 \\ any, \text{Должен послать } x + u, \text{ но } x \text{ неизвестен.}, c = 1 \end{cases}$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P' : c$
  5. $P' : r = \begin{cases} u, c = 0 \\ any, \text{Должен послать } x + u, \text{ но } x \text{ неизвестен.}, c = 1 \end{cases}$
  6. $P' \to V : r$

Почему V убеждается, что P знает секретный ключ x?

Покажем корректность протокола (soundness).

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 1**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 0}$. Тогда

1. $P' : u \in_R \mathbb{Z}_n,\ a = g^u$
2. $P' \to V : a$
3. $V : c \in_R \{0, 1\}$
4. $V \to P' : c$
5. $P' : r = \begin{cases} u, c = 0 \\ any, \text{Должен послать } x + u, \text{ но } x \text{ неизвестен.}, c = 1 \end{cases}$
6. $P' \to V : r$
7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah, c = 1 \end{cases}$

Почему V убеждается, что P знает секретный ключ x?

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда
    1. $P' : u \in_R \mathbb{Z}_n$, $a = \frac{g^u}{h}$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n$, $a = \frac{g^u}{h}$
  2. $P' \rightarrow V : a$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n,\ a = \frac{g^u}{h}$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n$, $a = \frac{g^u}{h}$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P' : c$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда
  1. $P' : u \in_R \mathbb{Z}_n$, $a = \frac{g^u}{h}$
  2. $P' \to V : a$
  3. $V : c \in_R \{0, 1\}$
  4. $V \to P' : c$
  5. $P' : r = \begin{cases} any, \text{Должен послать } u, \text{ но } g^u \neq a., c = 0 \\ u, c = 1 \end{cases}$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда

1. $P' : u \in_R \mathbb{Z}_n,\ a = \frac{g^u}{h}$
2. $P' \to V : a$
3. $V : c \in_R \{0, 1\}$
4. $V \to P' : c$
5. $P' : r = \begin{cases} any, \text{Должен послать } u, \text{ но } g^u \neq a., c = 0 \\ u, c = 1 \end{cases}$
6. $P' \to V : r$

Почему V убеждается, что P знает секретный ключ x?

- Пусть P' — атакующий, который не знает x, но хочет аутентифицироваться как P.
- **Вариант атаки 2**. Так как на шаге 3 значение c выбирается случайно, то с вероятностью $\frac{1}{2}$ атакующий P' до начала аутентификации может предположить, что появится $\underline{c = 1}$. Тогда

1. $P' : u \in_R \mathbb{Z}_n,\ a = \frac{g^u}{h}$
2. $P' \to V : a$
3. $V : c \in_R \{0, 1\}$
4. $V \to P' : c$
5. $P' : r = \begin{cases} any, \text{Должен послать } u, \text{ но } g^u \neq a., c = 0 \\ u, c = 1 \end{cases}$
6. $P' \to V : r$
7. $V : g^r \stackrel{?}{==} \begin{cases} a, c = 0 \\ ah = g^u/h \cdot h, c = 1 \end{cases}$

Почему V убеждается, что P знает секретный ключ x?

Почему V убеждается, что P знает секретный ключ x?

- Предположим, что атакующий P' может правильно ответить и на $c = 1$, и на $c = 0$, отправив перед этим анонс a.

Почему V убеждается, что P знает секретный ключ x?

- Предположим, что атакующий P' может правильно ответить и на $c = 1$, и на $c = 0$, отправив перед этим анонс a.
- Тогда P' может подготовить ответы r_0 (соответствует $c = 0$) и r_1 (соответствует $c = 1$), такие, что

$$g^{r_0} = a, \; g^{r_1} = a \cdot h.$$

Почему V убеждается, что P знает секретный ключ x?

- Предположим, что атакующий P' может правильно ответить и на $c = 1$, и на $c = 0$, отправив перед этим анонс a.
- Тогда P' может подготовить ответы r_0 (соответствует $c = 0$) и r_1 (соответствует $c = 1$), такие, что

$$g^{r_0} = a, \; g^{r_1} = a \cdot h.$$

- Отсюда получаем, что

$$h = g^{r_1 - r_0}.$$

Почему V убеждается, что P знает секретный ключ x?

- Предположим, что атакующий P' может правильно ответить и на $c = 1$, и на $c = 0$, отправив перед этим анонс a.
- Тогда P' может подготовить ответы r_0 (соответствует $c = 0$) и r_1 (соответствует $c = 1$), такие, что
$$g^{r_0} = a, \; g^{r_1} = a \cdot h.$$
- Отсюда получаем, что
$$h = g^{r_1 - r_0}.$$
- Другими словами, атакующий P' по h может найти его логарифм $\log_q h = r_1 - r_0$. Но предполагается, что проблема дискретного логарифма сложна для современной техники.

Почему V убеждается, что P знает секретный ключ x?

- Предположим, что атакующий P' может правильно ответить и на $c = 1$, и на $c = 0$, отправив перед этим анонс a.
- Тогда P' может подготовить ответы r_0 (соответствует $c = 0$) и r_1 (соответствует $c = 1$), такие, что
$$g^{r_0} = a, \; g^{r_1} = a \cdot h.$$
- Отсюда получаем, что
$$h = g^{r_1 - r_0}.$$
- Другими словами, атакующий P' по h может найти его логарифм $\log_q h = r_1 - r_0$. Но предполагается, что проблема дискретного логарифма сложна для современной техники.
- Поэтому предложение неверно!

Почему V убеждается, что P знает секретный ключ x?

Проблема атакующего

Атакующий во время получения запроса c не может подготовить правильный ответ и для $c = 0$, и для $c = 1$, не зная x. Он может только заранее подготовить ответ для одного из двух значений. И проблема в том, что это подобранное значение он сначала вручает (commit), а уже потом получает запрос c.

Почему V убеждается, что P знает секретный ключ x?

Проблема атакующего

Атакующий во время получения запроса c не может подготовить правильный ответ и для $c = 0$, и для $c = 1$, не зная x. Он может только заранее подготовить ответ для одного из двух значений. И проблема в том, что это подобранное значение он сначала вручает (commit), а уже потом получает запрос c.

Таким образом,

- вероятность пройти аутентификацию на **одной** итерации без знания ключа x равна $\frac{1}{2}$;

Почему V убеждается, что P знает секретный ключ x?

Проблема атакующего

Атакующий во время получения запроса c не может подготовить правильный ответ и для $c = 0$, и для $c = 1$, не зная x. Он может только заранее подготовить ответ для одного из двух значений. И проблема в том, что это подобранное значение он сначала вручает (commit), а уже потом получает запрос c.

Таким образом,

- вероятность пройти аутентификацию на **одной** итерации без знания ключа x равна $\frac{1}{2}$;
- вероятность пройти аутентификацию на **двух** итерациях без знания ключа x равна $\frac{1}{2} \cdot \frac{1}{2}$;

Почему V убеждается, что P знает секретный ключ x?

Проблема атакующего

Атакующий во время получения запроса c не может подготовить правильный ответ и для $c = 0$, и для $c = 1$, не зная x. Он может только заранее подготовить ответ для одного из двух значений. И проблема в том, что это подобранное значение он сначала вручает (commit), а уже потом получает запрос c.

Таким образом,

- вероятность пройти аутентификацию на **одной** итерации без знания ключа x равна $\frac{1}{2}$;
- вероятность пройти аутентификацию на **двух** итерациях без знания ключа x равна $\frac{1}{2} \cdot \frac{1}{2}$;
- вероятность пройти аутентификацию на **трех** итерациях без знания ключа x равна $\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2}$;

Почему V убеждается, что P знает секретный ключ x?

Проблема атакующего

Атакующий во время получения запроса c не может подготовить правильный ответ и для $c = 0$, и для $c = 1$, не зная x. Он может только заранее подготовить ответ для одного из двух значений. И проблема в том, что это подобранное значение он сначала вручает (commit), а уже потом получает запрос c.

Таким образом,

- вероятность пройти аутентификацию на **одной** итерации без знания ключа x равна $\frac{1}{2}$;
- вероятность пройти аутентификацию на **двух** итерациях без знания ключа x равна $\frac{1}{2} \cdot \frac{1}{2}$;
- вероятность пройти аутентификацию на **трех** итерациях без знания ключа x равна $\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2}$;
- ...
- вероятность пройти аутентификацию на **k** итерациях без знания ключа x равна $\frac{1}{2^k}$ (k – параметр безопасности, зависит от требований прикладной задачи).

ZK-свойство протокола Шнорра

Возможности атакующего

ZK-свойство протокола Шнорра

Возможности атакующего

- Атакующий может накапливать некоторое количество разговоров

$$\{(a_i, c_i, r_i) \in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n\}_{i=1}^{N}. \qquad (1)$$

ZK-свойство протокола Шнорра

Возможности атакующего

- Атакующий может накапливать некоторое количество разговоров

$$\{(a_i, c_i, r_i) \in \langle g \rangle \times \{0,1\} \times \mathbb{Z}_n\}_{i=1}^{N}. \tag{1}$$

- В качестве атакующего (без нарушения общности), можно рассматривать проверяющую сторону V. При этом

ZK-свойство протокола Шнорра

Возможности атакующего

- Атакующий может накапливать некоторое количество разговоров

$$\{(a_i, c_i, r_i) \in \langle g \rangle \times \{0,1\} \times \mathbb{Z}_n\}_{i=1}^{N}. \qquad (1)$$

- В качестве атакующего (без нарушения общности), можно рассматривать проверяющую сторону V. При этом
  - V может **честно** выполнять протокол (генерировать запросы c случайно и равновероятно)

ZK-свойство протокола Шнорра

Возможности атакующего

- Атакующий может накапливать некоторое количество разговоров

$$\{(a_i, c_i, r_i) \in \langle g \rangle \times \{0,1\} \times \mathbb{Z}_n\}_{i=1}^{N}. \qquad (1)$$

- В качестве атакующего (без нарушения общности), можно рассматривать проверяющую сторону V. При этом
  - V может **честно** выполнять протокол (генерировать запросы c случайно и равновероятно)
  - V может **нечестно** выполнять протокол (генерировать запросы c «как-то»)

ZK-свойство протокола Шнорра

Возможности атакующего

- Атакующий может накапливать некоторое количество разговоров

$$\{(a_i, c_i, r_i) \in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n\}_{i=1}^{N}. \tag{1}$$

- В качестве атакующего (без нарушения общности), можно рассматривать проверяющую сторону V. При этом
  - V может **честно** выполнять протокол (генерировать запросы c случайно и равновероятно)
  - V может **нечестно** выполнять протокол (генерировать запросы c «как-то»)

Цель

Если мы покажем, что атакующий может **сам** (без привлечения доказывающей стороны) сгенерировать (т.е. симулировать) разговоры (a', c', r'), которые распределены также, как и разговоры из (1), то тем самым докажем, что протокол Шнорра – это ZK-протокол.

ZK-свойство протокола Шнорра. Честный V

ZK-свойство протокола Шнорра. Честный V

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

$u \in_R \mathbb{Z}_n$
$a \leftarrow g^u$ (анонс)
$c \in_R \{0, 1\}$ (запрос)
$r \leftarrow u + c \cdot x$ (ответ)
вернуть (a, c, r)

ZK-свойство протокола Шнорра. Честный V

Реальный разговор	Симулир. разговор
Input: секретный ключ x **Output:** разговор (a, c, r) $u \in_R \mathbb{Z}_n$ $a \leftarrow g^u$ (анонс) $c \in_R \{0, 1\}$ (запрос) $r \leftarrow u + c \cdot x$ (ответ) вернуть (a, c, r)	**Input:** публичный ключ $h = g^x$ **Output:** разговор (a, c, r) $c \in_R \{0, 1\}$ («запрос») $r \in_R \mathbb{Z}_n$(«ответ») $a \leftarrow g^r h^{-c}$ («анонс») вернуть (a, c, r)

ZK-свойство протокола Шнорра. Честный V

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

1. $u \in_R \mathbb{Z}_n$
2. $a \leftarrow g^u$ (анонс)
3. $c \in_R \{0, 1\}$ (запрос)
4. $r \leftarrow u + c \cdot x$ (ответ)
5. вернуть (a, c, r)

Симулир. разговор

Input: публичный ключ $h = g^x$
Output: разговор (a, c, r)

1. $c \in_R \{0, 1\}$ («запрос»)
2. $r \in_R \mathbb{Z}_n$(«ответ»)
3. $a \leftarrow g^r h^{-c}$ («анонс»)
4. вернуть (a, c, r)

Наблюдения и выводы

ZK-свойство протокола Шнорра. Честный V

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

$u \in_R \mathbb{Z}_n$
$a \leftarrow g^u$ (анонс)
$c \in_R \{0, 1\}$ (запрос)
$r \leftarrow u + c \cdot x$ (ответ)
вернуть (a, c, r)

Симулир. разговор

Input: публичный ключ
$h = g^x$
Output: разговор (a, c, r)

$c \in_R \{0, 1\}$ («запрос»)
$r \in_R \mathbb{Z}_n$(«ответ»)
$a \leftarrow g^r h^{-c}$ («анонс»)
вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.

ZK-свойство протокола Шнорра. Честный V

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

$u \in_R \mathbb{Z}_n$
$a \leftarrow g^u$ (анонс)
$c \in_R \{0, 1\}$ (запрос)
$r \leftarrow u + c \cdot x$ (ответ)
вернуть (a, c, r)

Симулир. разговор

Input: публичный ключ $h = g^x$
Output: разговор (a, c, r)

$c \in_R \{0, 1\}$ («запрос»)
$r \in_R \mathbb{Z}_n$(«ответ»)
$a \leftarrow g^r h^{-c}$ («анонс»)
вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.
- Для любых $(A, C, R) \in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n$:

$$\Pr\{(a, c, r) = (A, C, R)\} = \frac{1}{2n}.$$

ZK-свойство протокола Шнорра. Честный V

Реальный разговор	Симулир. разговор
Input: секретный ключ x **Output:** разговор (a, c, r) 1 $u \in_R \mathbb{Z}_n$ 2 $a \leftarrow g^u$ (анонс) 3 $c \in_R \{0, 1\}$ (запрос) 4 $r \leftarrow u + c \cdot x$ (ответ) 5 вернуть (a, c, r)	**Input:** публичный ключ $h = g^x$ **Output:** разговор (a, c, r) 1 $c \in_R \{0, 1\}$ («запрос») 2 $r \in_R \mathbb{Z}_n$(«ответ») 3 $a \leftarrow g^r h^{-c}$ («анонс») 4 вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.
- Для любых $(A, C, R) \in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n$:

$$\Pr\{(a, c, r) = (A, C, R)\} = \frac{1}{2n}.$$

- Распределения одинаковые, как для реального разговора, так и для симулированного.

ZK-свойство протокола Шнорра. Честный V

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

$u \in_R \mathbb{Z}_n$
$a \leftarrow g^u$ (анонс)
$c \in_R \{0, 1\}$ (запрос)
$r \leftarrow u + c \cdot x$ (ответ)
вернуть (a, c, r)

Симулир. разговор

Input: публичный ключ $h = g^x$
Output: разговор (a, c, r)

$c \in_R \{0, 1\}$ («запрос»)
$r \in_R \mathbb{Z}_n$(«ответ»)
$a \leftarrow g^r h^{-c}$ («анонс»)
вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.
- Для любых $(A, C, R) \in \langle g \rangle \times \{0, 1\} \times \mathbb{Z}_n$:

$$\Pr\{(a, c, r) = (A, C, R)\} = \frac{1}{2n}.$$

- Распределения одинаковые, как для реального разговора, так и для симулированного.
- Если V — честный, то протокол Шнорра – ZK-протокол.

ZK-свойство протокола Шнорра. Нечестный V^*

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x

Output: разговор (a, c, r)

$u \in_R \mathbb{Z}_n$

$a \leftarrow g^u$

отправить a к V^* (анонс)

получить $c \in \{0, 1\}$ от V^* (запр.)

$r \leftarrow u + c \cdot x$

отправить r к V^* (ответ)

вернуть (a, c, r)

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)
1 $u \in_R \mathbb{Z}_n$
2 $a \leftarrow g^u$
3 отправить a к V^* (анонс)
4 получить $c \in \{0, 1\}$ от V^* (запр.)
5 $r \leftarrow u + c \cdot x$
6 отправить r к V^* (ответ)
7 вернуть (a, c, r)

Симулир. разговор

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)
1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить состояние V^* на шаг 4 и перейти к шагу 1
7 отправить r к V^*
8 вернуть (a, c, r)

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

1 $u \in_R \mathbb{Z}_n$
2 $a \leftarrow g^u$
3 отправить a к V^* (анонс)
4 получить $c \in \{0, 1\}$ от V^* (запр.)
5 $r \leftarrow u + c \cdot x$
6 отправить r к V^* (ответ)
7 вернуть (a, c, r)

Симулир. разговор

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить состояние V^* на шаг 4 и перейти к шагу 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

1 $u \in_R \mathbb{Z}_n$
2 $a \leftarrow g^u$
3 отправить a к V^* (анонс)
4 получить $c \in \{0, 1\}$ от V^* (запр.)
5 $r \leftarrow u + c \cdot x$
6 отправить r к V^* (ответ)
7 вернуть (a, c, r)

Симулир. разговор

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить состояние V^* на шаг 4 и перейти к шагу 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

1 $u \in_R \mathbb{Z}_n$
2 $a \leftarrow g^u$
3 отправить a к V^* (анонс)
4 получить $c \in \{0, 1\}$ от V^* (запр.)
5 $r \leftarrow u + c \cdot x$
6 отправить r к V^* (ответ)
7 вернуть (a, c, r)

Симулир. разговор

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить состояние V^* на шаг 4 и перейти к шагу 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.
- Распределения одинаковые, как для реального разговора, так и для симулированного (какие распределения – заранее неизвестно).

ZK-свойство протокола Шнорра. Нечестный V^*

Реальный разговор

Input: секретный ключ x
Output: разговор (a, c, r)

1 $u \in_R \mathbb{Z}_n$
2 $a \leftarrow g^u$
3 отправить a к V^* (анонс)
4 получить $c \in \{0, 1\}$ от V^* (запр.)
5 $r \leftarrow u + c \cdot x$
6 отправить r к V^* (ответ)
7 вернуть (a, c, r)

Симулир. разговор

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить состояние V^* на шаг 4 и перейти к шагу 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

- В обоих случаях разговоры (a, c, r) – допустимые. Т.е. ответ всегда соответствует запросу.
- Распределения одинаковые, как для реального разговора, так и для симулированного (какие распределения – заранее неизвестно).
- Если V — нечестный, то протокол Шнорра – также ZK-протокол.

ZK-свойство протокола Шнорра. Выводы.

ZK-свойство протокола Шнорра. Выводы.

Вывод 1

Независимо от стратегии мошенничества V, протокол Шнорра — это ZK-протокол.

ZK-свойство протокола Шнорра. Выводы.

Вывод 1

Независимо от стратегии мошенничества V, протокол Шнорра — это ZK-протокол.

Вывод 2

Перехват разговоров не имеет смысла для атакующего. Остаются только атаки на ключ h.

Нестаток протокола Шнорра

Недостаток рассмотренной версии протокола

Для параметра безопасности k протокол выполняется k раз (обеспечивая вероятность успешной аутентификации 2^{-k} без знания ключа). На каждой итерации требуется выполнять возведение в степень в группе порядка n — «дорогостоящая» операция.

Нестаток протокола Шнорра

Недостаток рассмотренной версии протокола

Для параметра безопасности k протокол выполняется k раз (обеспечивая вероятность успешной аутентификации 2^{-k} без знания ключа). На каждой итерации требуется выполнять возведение в степень в группе порядка n — «дорогостоящая» операция.

Шнорр предложил протокол с **одной итерацией**. Этот протокол обычно и принято называть *протоколом Шнорра*.

Протокол Шнорра с одной итерацией

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.

Протокол Шнорра с одной итерацией

- Пусть $\langle g \rangle$ — циклическая группа с порождающим элементом g, $n = |\langle g \rangle|$ – порядок этой группы (большое, желательно простое число). **Ключи доказывающей стороны**:
  - $sk_P = x$, $x \in_R \mathbb{Z}_n$ — случайно выбранный из группы элемент,
  - $pk_P = g^x$. Этот ключ передается проверяющей стороне в фазе регистрации.
-
  1. $P : u \in_R \mathbb{Z}_n$, $a = g^u$
  2. $P \to V : a$ — вручение обязательства (прячется ключ u)
  3. $V : c \in_R \mathbb{Z}_n$
  4. $V \to P : c$
  5. $P : r = u + x \cdot c$
  6. $P \to V : r$ — «раскрытие обязательства»
  7. $V : g^r \stackrel{?}{==} a \cdot h^c$

Корректность (soundness) протокола Шнорра

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.
- Тогда $h = g^{(r-r')/(c-c')}$.

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.
- Тогда $h = g^{(r-r')/(c-c')}$.
- Получаем, что атакующий P' может **эффективно** вычислить $\log_g h = \frac{r-r'}{c-c'}$.

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.
- Тогда $h = g^{(r-r')/(c-c')}$.
- Получаем, что атакующий P' может **эффективно** вычислить $\log_g h = \frac{r-r'}{c-c'}$.
- Но на сегодняшний момент нет таких алгоритмов, которые бы эффективно находили логарифм случайного элемента циклической группы (помним, что $h = g^x$ — случайный элемент группы $\langle g \rangle$).

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.
- Тогда $h = g^{(r-r')/(c-c')}$.
- Получаем, что атакующий P' может **эффективно** вычислить $\log_g h = \frac{r-r'}{c-c'}$.
- Но на сегодняшний момент нет таких алгоритмов, которые бы эффективно находили логарифм случайного элемента циклической группы (помним, что $h = g^x$ — случайный элемент группы $\langle g \rangle$).
- **Значит наше предположение неверно**!

Корректность (soundness) протокола Шнорра

- Предположим, что атакующий P', который не знает секретный ключ $sk_P = x$, может правильно (и **эффективно**) ответить на **хотя бы два** возможных запроса c и c', после того, как отправил анонс a.
- Это означает, что P' может подготовить два правильных разговора: (a, c, r) и (a, c', r').
- Откуда получаем: $g^r = ah^c$, $g^{r'} = ah^{c'}$.
- Тогда $h = g^{(r-r')/(c-c')}$.
- Получаем, что атакующий P' может **эффективно** вычислить $\log_g h = \frac{r-r'}{c-c'}$.
- Но на сегодняшний момент нет таких алгоритмов, которые бы эффективно находили логарифм случайного элемента циклической группы (помним, что $h = g^x$ — случайный элемент группы $\langle g \rangle$).
- **Значит наше предположение неверно**!

Вывод

Атакующий может ответить правильно только на один запрос (какой при этом будет анонс?). Так как запрос выбирается случайно из $\mathbb{Z}_n$, а n, по предположению, очень большое ($n \geqslant 2^{128}$), то вероятность угадать запрос — $1/n$.

ZK-свойство протокола Шнорра. Честный V

ZK-свойство протокола Шнорра. Честный V

- При взаимодействии с P проверяющая сторона V (или пассивный наблюдатель) может **накопить** набор разговоров:

$$\{(a, c, r) : u, c \in_R \mathbb{Z}_n, r = u + c \cdot x\}.$$

ZK-свойство протокола Шнорра. Честный V

- При взаимодействии с P проверяющая сторона V (или пассивный наблюдатель) может **накопить** набор разговоров:

$$\{(a, c, r) : u, c \in_R \mathbb{Z}_n, r = u + c \cdot x\}.$$

- Но и без взаимодействия с P проверяющая сторона V может **построить** набор разговоров:

$$\{(a, c, r) : c, r \in_R \mathbb{Z}_n, a = g^r h^{-c}\}.$$

ZK-свойство протокола Шнорра. Честный V

- При взаимодействии с P проверяющая сторона V (или пассивный наблюдатель) может **накопить** набор разговоров:

$$\{(a, c, r) : u, c \in_R \mathbb{Z}_n, r = u + c \cdot x\}.$$

- Но и без взаимодействия с P проверяющая сторона V может **построить** набор разговоров:

$$\{(a, c, r) : c, r \in_R \mathbb{Z}_n, a = g^r h^{-c}\}.$$

Эти распределения одинаковые: каждое значение (тройка (A, B, C)) появляется с вероятностью $\frac{1}{n^2}$.

ZK-свойство протокола Шнорра. Честный V

- При взаимодействии с P проверяющая сторона V (или пассивный наблюдатель) может **накопить** набор разговоров:

$$\{(a, c, r) : u, c \in_R \mathbb{Z}_n, r = u + c \cdot x\}.$$

- Но и без взаимодействия с P проверяющая сторона V может **построить** набор разговоров:

$$\{(a, c, r) : c, r \in_R \mathbb{Z}_n, a = g^r h^{-c}\}.$$

Эти распределения одинаковые: каждое значение (тройка (A, B, C)) появляется с вероятностью $\frac{1}{n^2}$.

Вывод

Честный проверяющий или пассивный атакующий не получат какую-либо информацию из разговоров с доказывающей стороной.

ZK-свойство протокола Шнорра. Нечестный V^*.

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0, 1\}$

Input: публ. ключ $h = g^x$

Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить V^*
   на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0,1\}$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0,1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0,1\}$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Симулир. разговор $c \in \mathbb{Z}_n$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0, 1\}$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Симулир. разговор $c \in \mathbb{Z}_n$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0, 1\}$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)
1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Симулир. разговор $c \in \mathbb{Z}_n$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)
1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

- Симулятор для протокола Шнорра с одной итерацией будет работать **неполиномиально долго**.

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0, 1\}$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \{0, 1\}$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Симулир. разговор $c \in \mathbb{Z}_n$

Input: публ. ключ $h = g^x$
Output: разговор (a, c, r)

1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*
5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*
8 вернуть (a, c, r)

Наблюдения и выводы

- Симулятор для протокола Шнорра с одной итерацией будет работать **неполиномиально долго**.
- Поэтому нельзя сказать, что нечестный V^* может симулировать разговоры.

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0,1\}$	Симулир. разговор $c \in \mathbb{Z}_n$
Input: публ. ключ $h = g^x$	**Input:** публ. ключ $h = g^x$
Output: разговор (a, c, r)	**Output:** разговор (a, c, r)
1 $c \in_R \{0,1\}$ («запрос»)	1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)	2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)	3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*	4 отправить a к V^*
5 получить $c' \in \{0,1\}$ от V^*	5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1	6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*	7 отправить r к V^*
8 вернуть (a, c, r)	8 вернуть (a, c, r)

Наблюдения и выводы

- Симулятор для протокола Шнорра с одной итерацией будет работать **неполиномиально долго**.
- Поэтому нельзя сказать, что нечестный V^* может симулировать разговоры.
- Поэтому нельзя утверждать, что протокол Шнорра с одной итерацией обладает ZK-свойством.

ZK-свойство протокола Шнорра. Нечестный V^*.

Сим. разг., $c \in \{0, 1\}$	Симулир. разговор $c \in \mathbb{Z}_n$
Input: публ. ключ $h = g^x$	**Input:** публ. ключ $h = g^x$
Output: разговор (a, c, r)	**Output:** разговор (a, c, r)
1 $c \in_R \{0, 1\}$ («запрос»)	1 $c \in_R \mathbb{Z}_n$ («запрос»)
2 $r \in_R \mathbb{Z}_n$(«ответ»)	2 $r \in_R \mathbb{Z}_n$(«ответ»)
3 $a \leftarrow g^r h^{-c}$ («анонс»)	3 $a \leftarrow g^r h^{-c}$ («анонс»)
4 отправить a к V^*	4 отправить a к V^*
5 получить $c' \in \{0, 1\}$ от V^*	5 получить $c' \in \mathbb{Z}_n$ от V^*
6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1	6 если $c' \neq c$ то откатить V^* на ш. 4 и перейти к ш. 1
7 отправить r к V^*	7 отправить r к V^*
8 вернуть (a, c, r)	8 вернуть (a, c, r)

Наблюдения и выводы

- Симулятор для протокола Шнорра с одной итерацией будет работать **неполиномиально долго**.
- Поэтому нельзя сказать, что нечестный V^* может симулировать разговоры.
- Поэтому нельзя утверждать, что протокол Шнорра с одной итерацией обладает ZK-свойством.
- Но тем не менее пока нет эффективных атак на этот протокол.

Домашнее задание.

- Изучить протокол Гиллу-Кискате[1].
- Изучить протокол Штерна[2].
- Изучить протокол Окамото[3].

[1] L. C. Guillou and J.-J. Quisquater. A practical zero-knowledge protocol fitted to security microprocessor minimizing both transmission and memory. In Advances in Cryptology|EUROCRYPT '88, volume 330 of LNCS, pages 123-128, Berlin, 1988. Springer.

[2] Stern J. (1994) A new identification scheme based on syndrome decoding. In: Stinson D.R. (eds) Advances in Cryptology — CRYPTO' 93. CRYPTO 1993. Lecture Notes in Computer Science, vol 773. Springer, Berlin, Heidelberg.

[3] T. Okamoto. Provably secure and practical identifcation schemes and corresponding signature schemes. In Advances in Cryptology|CRYPTO '92, volume 740 of LNCS, pages 31-53, Berlin, 1993. Springer.

Заключение

Спасибо за внимание!