Cryptography

Lecture 5: Pseudo One-Time Pad

November 13, 2026

Pseudorandom Generator

§4.5 Observation

If P = NP then pseudorandom generators cannot exist

Proof

Note that universal distinguisher of previous slide is in NP.
If P = NP, then it is polynomial-time and thus 2nd item of Definition §4.4 cannot be fulfilled. □

Pseudorandom Generator

§4.6 Definition

Given polynomial p, p-ensemble is function $\left(\mathfrak{X}_n \colon \{0,1\}^{p(n)} \to [0,1]\right)_{n\in\mathbb{N}}$ such that $\mathfrak{X}_n$ is probability distribution for all $n \in \mathbb{N}$.

p-ensembles $\mathfrak{X}$ and $\mathfrak{Y}$ are computationally indistinguishable, written $\mathfrak{X} \equiv_{\text{PPT}} \mathfrak{Y}$, if for every PPT algorithm $D \colon \{1\}^* \times \{0,1\}^* \to \{0,1\}$

$$\mathrm{E}\Big[\mathrm{P}\big(D(1^n, x)\big)\Big]_{x\leftarrow\mathfrak{X}_n} \simeq \mathrm{E}\Big[\mathrm{P}\big(D(1^n, y)\big)\Big]_{y\leftarrow\mathfrak{Y}_n}$$

Notes

- Success probabilities of algorithms $x \leftarrow \mathfrak{X}_n\,;\, D(1^n, x)$ and $y \leftarrow \mathfrak{Y}_n\,;\, D(1^n, y)$ asymptotically almost equal
- 2nd item of §4.4 simplifies to $\left(\frac{|G^{-1}(r)|}{2^n}\right)_{r\in U_{p(n)}} \equiv_{\text{PPT}} U_{p(n)}$
  (pseudorandom = computationally indistinguishable from uniform)

Pseudorandom Generator

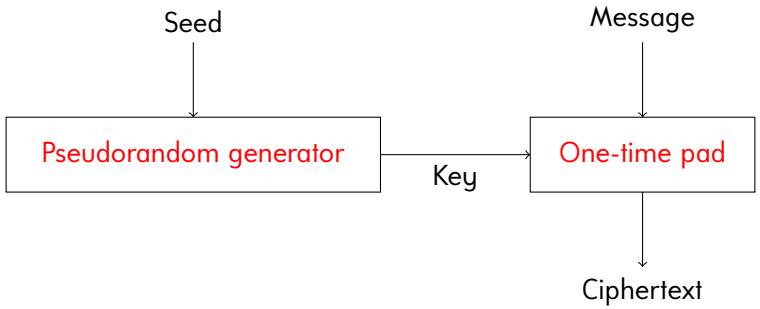
Questions

- Is $\equiv_{\text{PPT}}$ equivalence relation?
- Does P ≠ NP imply existence of pseudorandom generator?
  (converse of Observation §4.5; worst- vs. average-case complexity)
  Careful: Would make cryptographers very happy if you prove this
- Are pseudorandom generators closed under composition?
  (Given pseudorandom generators G_1 and G_2 is G given by $G(s) = G_2(G_1(s))$ for all $s \in \{0,1\}^*$ again pseudorandom generator?)

EAV-Secure Cipher

Motivation

- One-time pad needs long random keys
- Pseudorandom generator can turn short random key (seed) into long pseudorandom key for one-time pad (that is computationally indistinguishable from random)
- Combination may solve long key problem



EAV-Secure Cipher

§4.7 Construction

Let G be pseudorandom generator with expansion p.
Define private-key encryption scheme $\mathcal{E}_G = (\text{gen}, \text{enc}, \text{dec})$ with

- $P_k^n(k) = 2^{-n}$ for all $n \in \mathbb{N}$ and $k \in \{0,1\}^n$
- $\text{enc}_k(m) = G(k) \oplus m$ for all $k \in K$ and $m \in M$ with $|m| = p(|k|)$
- $\text{dec} = \text{enc}$

Notes

- Can only encrypt messages m of particular lengths $|m| \in \text{ran}(p)$
- Keys can be much shorter than messages; i.e. $n \ll p(n)$

Multiple Encryptions

Summary

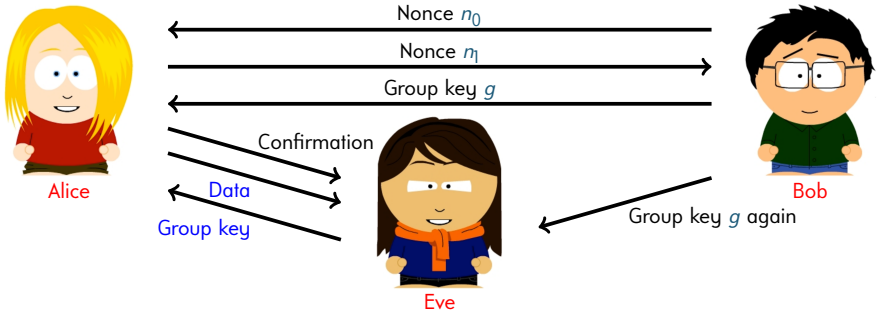
- One-time pad perfectly secret
- Pseudo one-time pad (Construction §4.7) EAV-secure but requires pseudorandom generator
- Security guarantee applies to key used once
- Both terrible in other threat models (in particular: key reuse)

Famous "two-time pads"

- Venona project (1943–1980): Counterintelligence program by NSA (broke 3,000 messages due to reused key pages)
- MS-PPTP using MS-CHAP-v1 (all Windows versions since 98 and NT) (pseudo one-time pad with same seed for communication Client → Server & Server → Client)
- IEEE 802.11 WEP (1997): WLAN encryption standard (one-time pad keys repeat after 2^{24} frames ≈ 5h with high traffic)

Multiple Encryptions

KRACK (Key Reinstallation Attack, 2017) against WPA2



- Man-in-the-middle attack
- After receiving g Alice confirms & starts data exchange
- Eve intercepts confirmation which triggers Bob to resend g
- Eve delays resent g until she has enough data
- Resent g causes Alice to reset & next data uses same key

EAV-Secure Cipher

§5.1 Theorem

Scheme $\mathcal{E}_G$ is EAV-secure for every pseudorandom generator G
(If G is pseudorandom generator, then scheme $\mathcal{E}_G$ is EAV-secure)

Proof (1/2)

By contraposition and reduction: Suppose that $\mathcal{E}_G$ is not EAV-secure, so there exists PPT algorithm $\mathcal{A}$ such that $\mathrm{P}\left[\mathrm{PrivK}^{\mathrm{one}}_{\mathcal{E}_G,\mathcal{A}}(n)\right] \not\approx \frac{1}{2}$.

We construct distinguisher D with inputs 1^n and $w \in \{0,1\}^{p(n)}$ as follows:

1. Run $\mathcal{A}(1^n)$ to obtain messages $m_0, m_1 \in \{0,1\}^{p(n)}$
2. $b \leftarrow U_1$ and set $c = m_b \oplus w$
3. Let $b' \leftarrow \mathcal{A}(c)$ and return 1 if $b' = b$ and return 0 otherwise

Clearly D is PPT algorithm since $\mathcal{A}$ is.

EAV-Secure Cipher

Proof (2/2)

We claim that D distinguishes $\left(\frac{|G^{-1}(r)|}{2^n}\right)_{r\in U_{p(n)}}$ from $U_{p(n)}$.

Let $\mathcal{E}_\ell$ denote one-time pad of length ℓ. Algorithm $r \leftarrow U_{p(n)}\,;\, D(1^n, r)$ is essentially $\mathrm{PrivK}^{\mathrm{one}}(\mathcal{E}_{p(n)}, \mathcal{A})$, so by Theorems 2.9 and 3.3

$$\mathrm{E}[\mathrm{P}(D(1^n, r))]_{r\leftarrow U_{p(n)}} = \mathrm{P}[\mathrm{PrivK}^{\mathrm{one}}(\mathcal{E}_{p(n)}, \mathcal{A})] = \frac{1}{2}$$

On the other hand, algorithm $s \leftarrow U_n\,;\, r = G(s)\,;\, D(1^n, r)$ is $\mathrm{PrivK}^{\mathrm{one}}_{\mathcal{E}_G,\mathcal{A}}(n)$, so

$$\mathrm{E}[\mathrm{P}(D(1^n, G(s)))]_{s\leftarrow U_n} = \mathrm{P}[\mathrm{PrivK}^{\mathrm{one}}_{\mathcal{E}_G,\mathcal{A}}(n)] \not\approx \frac{1}{2} = \mathrm{E}[\mathrm{P}(D(1^n, r))]_{r\leftarrow U_{p(n)}}$$

Hence $\left(\frac{|G^{-1}(r)|}{2^n}\right)_{r\in U_{p(n)}} \not\approx_{\mathrm{PPT}} U_{p(n)}$ and G is no pseudorandom generator. □

Stream Cipher

Optimizations

- Encrypt shorter messages by discarding superfluous bits from pseudorandom generator
- Avoid generating those superfluous bits in stream cipher

Stream cipher

- Generates bits one-by-one (or in larger blocks)
- Keeps state to remember current configuration
- Since pseudorandom generator is computable, there exists Turing machine (TM) that generates output sequentially (as in "Complexity theory" lecture)
  → Once output produced, stop TM & return (bit, configuration)
  → Next call is provided configuration & continues computation

Stream Cipher

§5.2 Definition (Stream cipher)

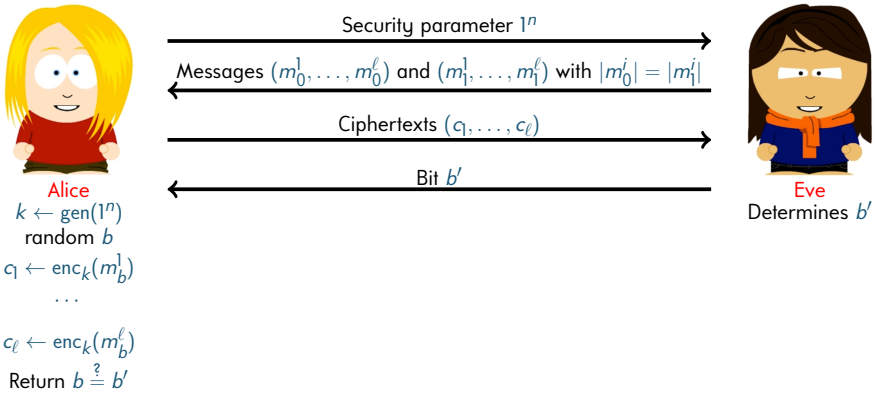
Stream cipher is pair (init, next) of deterministic algorithms

- $\mathrm{init}\colon \underbrace{\{0,1\}^*}_{\text{seed}} \times \underbrace{\{0,1\}^*}_{\text{initial vector}} \to \underbrace{\{0,1\}^*}_{\text{initial state}}$
  takes seed & initial vector; returns initial state
- $\mathrm{next}\colon \underbrace{\{0,1\}^*}_{\text{current state}} \to \underbrace{\{0,1\}^*}_{\text{output}} \times \underbrace{\{0,1\}^*}_{\text{next state}}$
  takes current state; returns output bits & next state

Stream cipher operation: Input seed s, initial vector v, rounds ℓ

1. $s_0 = \mathrm{init}(s, v)$
2. for $i = 1$ to ℓ: $\quad (b_i, s_i) = \mathrm{next}(s_{i-1})$
3. return $(b_1, \ldots, b_\ell)$

Multiple Encryptions



Notes

- Adversary provides two lists of messages
- Alice encrypts one of the lists & asks adversary which was encrypted

Multiple Encryptions

§5.3 Definition (Multiple-encryption indistinguishability game)

Let $n \in \mathbb{N}$, $\mathcal{E} = (\text{gen}, \text{enc}, \text{dec})$ efficient private-key encryption scheme and $\mathcal{A}$ stateful algorithm. Multiple-encryption adversarial indistinguishability game $\text{PrivK}^{\text{many}}_{\mathcal{E},\mathcal{A}}(n)$ is

1. Security parameter 1^n sent to adversary $\mathcal{A}$
2. $\mathcal{A}$ selects $\ell \in \mathbb{N}$, and message lists $(m_0^1, \ldots, m_0^\ell) \in \mathcal{M}^\ell$ and $(m_1^1, \ldots, m_1^\ell) \in \mathcal{M}^\ell$ with $|m_0^i| = |m_1^i|$ for all $1 \leq i \leq \ell$
3. $k \leftarrow \text{gen}(1^n)$; $b \leftarrow U_1$; $c_1 \leftarrow \text{enc}_k(m_b^1), \ldots, c_\ell \leftarrow \text{enc}_k(m_b^\ell)$ ciphertext list $(c_1, \ldots, c_\ell)$ sent to $\mathcal{A}$ (challenge ciphertexts)
4. $\mathcal{A}$ outputs bit $b' \in \{0,1\}$
5. Return 1 if $b = b'$ ($\mathcal{A}$ wins)
   Return 0 otherwise

Multiple Encryptions

§5.4 Definition (Multiple-encryption EAV-secure)

Private-key encryption scheme $\mathcal{E} = (\text{gen}, \text{enc}, \text{dec})$ is multiple-encryption EAV-secure if for every PPT algorithm $\mathcal{A}$

$$\text{P}\left[\text{PrivK}^{\text{many}}_{\mathcal{E},\mathcal{A}}(n)\right] \simeq \frac{1}{2}$$

Notes

- Surprisingly strong requirement (as we will see)
- Rather irrelevant in practice

Multiple Encryptions

Example: "Two-time pad"

- On receipt of n, adversary selects messages $(0^n, 0^n)$ & $(0^n, 1^n)$
- For any key k Alice returns either

$$\underbrace{(0^n \oplus k, 0^n \oplus k)}_{(k,k)} \quad \text{or} \quad \underbrace{(0^n \oplus k, 1^n \oplus k)}_{(k,\bar{k})}$$

- Adversary receives (c_1, c_2) and returns 0 if $c_1 = c_2$ and 1 otherwise
- Adversary always wins