

# Private-Key Encryption and Pseudorandomness (Part I)

Yu Zhang

HIT/CST/NIS

Cryptography, Autumn, 2014

- 1 A Computational Approach to Cryptography**
- 2 Defining Computationally-Secure Encryption**
- 3 Pseudorandomness**
- 4 Constructing Secure Encryption Schemes**

## **1** A Computational Approach to Cryptography

## 2 Defining Computationally-Secure Encryption

## 3 Pseudorandomness

## 4 Constructing Secure Encryption Schemes

# Idea of Computational Security

Computational security vs. Information-theoretical security

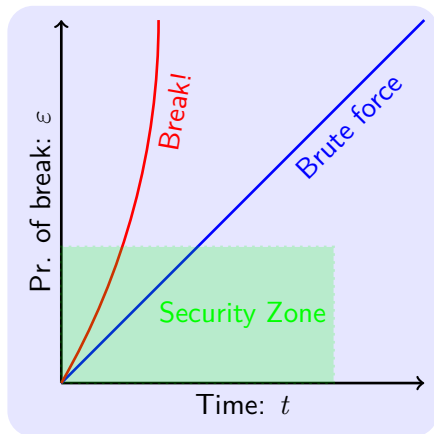
## Kerckhoffs's Another Principle

A [cipher] must be practically, if not mathematically, indecipherable.

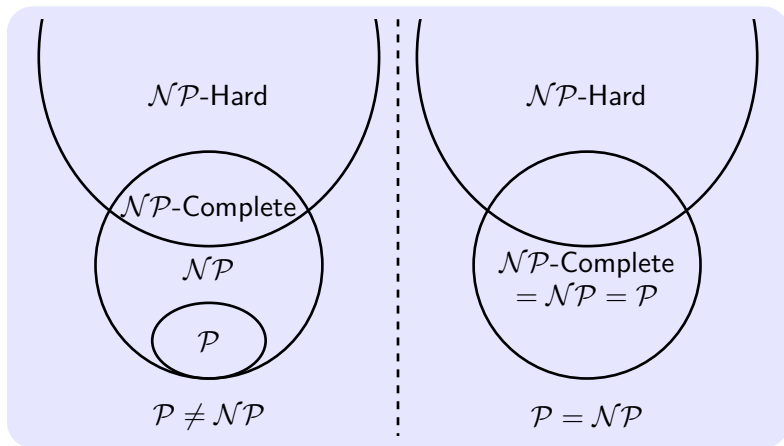
- Information-theoretical security: Perfect secrecy.  
Q: what's the limitation of perfect secrecy?
- Computational security:
  - Only preserved against adversaries that run in a **feasible amount of time**.
  - Adversaries can succeed with some **very small probability**.

# Necessity of the Relaxations

Limit the power of adversary (against brute force with pr. 1 in time linear in  $|\mathcal{K}|$ ) and allow a negligible probability (against random guess with pr.  $1/|\mathcal{K}|$ ).



$$\mathcal{P} = \mathcal{NP} ?$$



The majority of computer scientists believe  $\mathcal{P} \neq \mathcal{NP}$ .

*This is very dangerous!*

# Efficient Computation

- An algorithm  $A$  runs in **polynomial time** if there exists a polynomial  $p(\cdot)$  such that, for every input  $x \in 0, 1^*$ ,  $A(x)$  terminates within at most  $p(|x|)$  steps.

Q: is  $n!$  polynomial? is  $\log n$  polynomial?

- $A$  can run another PPT  $A'$  as a sub-routine in polynomial-time.

Q:  $f(x) = x^2$ , is  $g(x) = \frac{x^3}{f(x)}$  polynomial?

- A **probabilistic** algorithm has the capability of “tossing coins”. Random number generators should be designed for cryptographic use, not `random()` in C.
- Open question: Does probabilistic adversaries are more powerful than deterministic ones?

# Negligible Success Probability

- A function  $f$  is **negligible** if for every polynomial  $p(\cdot)$  there exists an  $N$  such that for all integers  $n > N$  it holds that  $f(n) < \frac{1}{p(n)}$ .
  - Q: is  $\left(\frac{3}{n}\right)^9$  negligible? is  $\frac{n^2}{2^n}$  negligible?
- Q: is  $\text{negl}_1(n) + \text{negl}_2(n)$  negligible?
- Q: is  $\text{poly}(n) \cdot \text{negl}(n)$  negligible?



# Asymptotic Approach

Problem X (breaking the scheme) is *hard* if X cannot be solved by any polynomial-time algorithm for time  $t$  except with negligible probability  $\varepsilon$ .

- $t, \varepsilon$  are described as functions of **security parameter**  $n$  (usually, the length of key).
- **Caution:** ‘Security’ for large enough values of  $n$ .

## Example

“Breaking the scheme” with probability  $2^{40} \cdot 2^{-n}$  in  $n^3$  minutes.

$n \leq 40$       6 weeks with probability 1.

$n = 50$       3 months with probability  $1/1000$ .

$n = 500$     more than 200 years with probability  $2^{-500}$ .

**Q: What if under Moore's Law?**

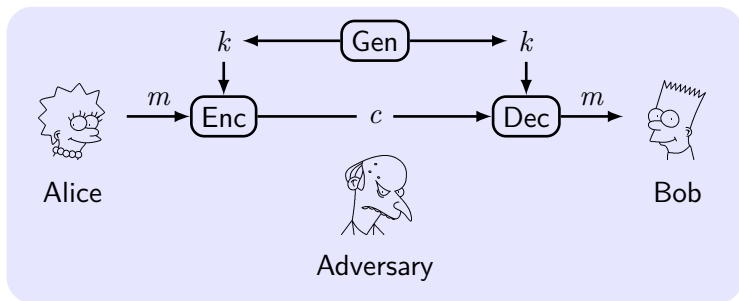
1 A Computational Approach to Cryptography

**2 Defining Computationally-Secure Encryption**

3 Pseudorandomness

4 Constructing Secure Encryption Schemes

# Defining Private-key Encryption Scheme



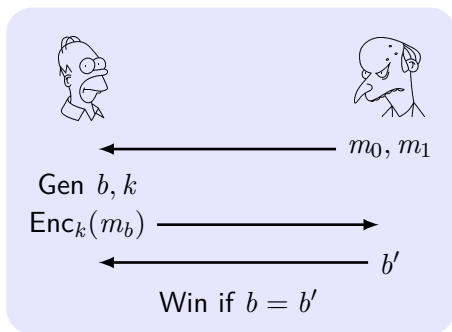
A **Private-key encryption scheme**  $\Pi$  is a tuple of PPT  $(\text{Gen}, \text{Enc}, \text{Dec})$

- $k \leftarrow \text{Gen}(1^n), |k| \geq n$  (security parameter).  
 $\text{Gen}(1^n)$  chooses  $k \leftarrow \{0, 1\}^n$  uniformly at random (**u.a.r.**).
- $c \leftarrow \text{Enc}_k(m), m \in \{0, 1\}^*$  (all finite-length binary strings).  
**Fixed-length** if  $m \in \{0, 1\}^{\ell(n)}$ .
- $m := \text{Dec}_k(c)$ .
- $\text{Dec}_k(\text{Enc}_k(m)) = m$ .

# Eavesdropping Indistinguishability Experiment

The eavesdropping indistinguishability experiment  $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}(n)$ :

- 1  $\mathcal{A}$  is given input  $1^n$ , outputs  $m_0, m_1$  of the same length.
- 2  $k \leftarrow \text{Gen}(1^n)$ , a random bit  $b \leftarrow \{0, 1\}$  is chosen. Then  $c \leftarrow \text{Enc}_k(m_b)$  (challenge ciphertext) is given to  $\mathcal{A}$ .
- 3  $\mathcal{A}$  outputs  $b'$ . If  $b' = b$ ,  $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}} = 1$ , otherwise 0.



# Defining Private-key Encryption Security

## Definition 1

$\Pi$  has **indistinguishable encryptions in the presence of an eavesdropper** if  $\forall$  PPT  $\mathcal{A}$ ,  $\exists$  a negligible function  $\text{negl}$  such that

$$\Pr \left[ \text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1 \right] \leq \frac{1}{2} + \text{negl}(n),$$

where the probability is taken over the random coins used by  $\mathcal{A}$ .

Q: Is the OTP scheme indistinguishable in the presence of an eavesdropper?

# Understanding Definition of Indistinguishability

**If the lowest bit of  $m$  can be guessed from the ciphertext with probability  $\frac{3}{4}$ , is it secure?**

Q: what are two messages provided by the adversary?

Q: what is the probability of success in this indistinguishability experiment?

**If the lowest 3 bits can be guessed with probability  $\frac{3}{8}$ , is it secure?**

**Intuition:** No partial information leaks.

## Definition 2

$\Pi$  is **semantically secure in the presence of an eavesdropper** if  $\forall$  PPT  $\mathcal{A}$ ,  $\exists \mathcal{A}'$  such that  $\forall$  distribution  $X = (X_1, \dots)$  and  $\forall f, h$ ,

$$|\Pr[\mathcal{A}(1^n, \text{Enc}_k(m), h(m)) = f(m)] - \Pr[\mathcal{A}(1^n, h(m)) = f(m)]| \\ \leq \text{negl}(n).$$

where  $m$  is chosen according to  $X_n$ ,  $h(m)$  is external information.

## Theorem 3

*A private-key encryption scheme has **indistinguishable** encryptions in the presence of an eavesdropper  $\iff$  it is **semantically secure** in the presence of an eavesdropper.*

- 1 A Computational Approach to Cryptography
- 2 Defining Computationally-Secure Encryption
- 3 Pseudorandomness**
- 4 Constructing Secure Encryption Schemes



# Conceptual Points of Pseudorandomness

- True randomness can not be generated by a describable mechanism.
- Pseudorandom looks truly random for the observers who don't know the mechanism.
- No fixed string can be “pseudorandom” which refers to a distribution.
- Q: is it possible to definitively prove randomness?



# Distinguisher: Statistical Tests

The pragmatic approach is to take many sequences of random numbers from a given generator and subject them to a battery of statistical tests.<sup>1</sup>

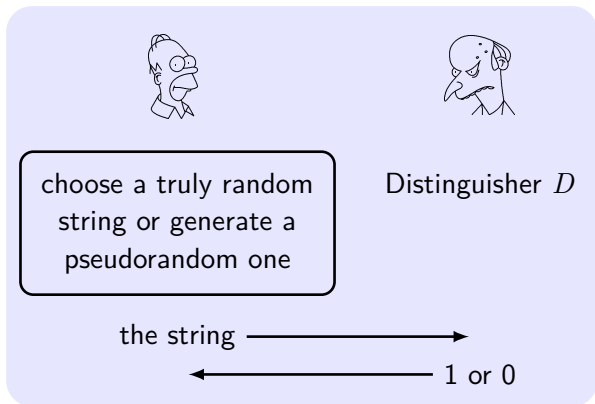
- $D(x) = 0$  if  $|\#0(x) - \#1(x)| \leq 10 \cdot \sqrt{n}$
- $D(x) = 0$  if  $|\#00(x) - n/4| \leq 10 \cdot \sqrt{n}$
- $D(x) = 0$  if  $\text{max-run-of-0}(x) \leq 10 \cdot \log n$

---

<sup>1</sup>State-of-the-art: NIST Special Publication 800-22 “A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications”

# Intuition for Defining Pseudorandom

**Intuition:** Generate a long string from a short truly random seed, and the pseudorandom string is indistinguishable from truly random strings.



# Definition of Pseudorandom Generators

## Definition 4

A deterministic polynomial-time algorithm  $G : \{0, 1\}^n \rightarrow \{0, 1\}^{\ell(n)}$  is a **pseudorandom generator (PRG)** if

- 1 (Expansion:):  $\forall n, \ell(n) > n$ .
- 2 (Pseudorandomness):  $\forall$  PPT distinguishers  $D$ ,

$$|\Pr[D(r) = 1] - \Pr[D(G(s)) = 1]| \leq \text{negl}(n),$$

where  $r$  is chosen *u.a.r* from  $\{0, 1\}^{\ell(n)}$ , the **seed**  $s$  is chosen *u.a.r* from  $\{0, 1\}^n$ .  $\ell(\cdot)$  is the **expansion factor** of  $G$ .

- Pseudorandomness means being **Unpredictable**
- **Existence:** Under the weak assumption that *one-way functions* exists, or  $\mathcal{P} \neq \mathcal{NP}$

## Is $G$ PRG?

- $G : s \rightarrow \{0, 1\}^n$  is such that for all  $s$ :  $XOR(G(s)) = 1$
- `glibc random()`:  $r[i] = (r[i - 3] + r[i - 31]) \% 2^{32}$

## $F$ is PRG. Is $G$ PRG?

- $G(s) = F(s) \oplus 1^n$
- $G(s) = F(0)$
- $G(s) = F(s) \| 0$
- $G(s) = F(s \oplus 1^{|s|})$
- $G(s) = F(s) \| F(s)$
- $G(s \| s') = F(s) \| F(s')$
- $G : s \leftarrow \{0, 1\}^{20}, G(s) = F(s)$  (see next slide)

# Sufficient seed space

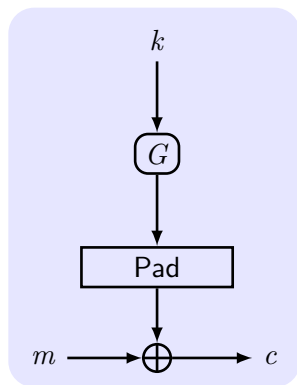
- **Sparse outputs:** In the case of  $\ell(n) = 2n$ , only  $2^{-n}$  of strings of length  $2n$  occurs.
- **Brute force attack:** Given an unlimited amount of time, one can distinguish  $G(s)$  from  $r$  with a high probability by generating all strings with all seeds.

$$|\Pr[D(r) = 1] - \Pr[D(G(s)) = 1]| \geq 1 - 2^{-n}$$

- **Sufficient seed space:**  $s$  must be long enough against brute force attack.

- 1 A Computational Approach to Cryptography
- 2 Defining Computationally-Secure Encryption
- 3 Pseudorandomness
- 4 Constructing Secure Encryption Schemes**

# A Secure Fixed-Length Encryption Scheme



## Construction 5

- $|G(k)| = \ell(|k|)$ ,  $m \in \{0, 1\}^{\ell(n)}$ .
- Gen:  $k \in \{0, 1\}^n$ .
- Enc:  $c := G(k) \oplus m$ .
- Dec:  $m := G(k) \oplus c$ .

## Theorem 6

*This fixed-length encryption scheme has indistinguishable encryptions in the presence of an eavesdropper.*



# Reduction (Complexity)

A **reduction** is a transformation of one problem  $A$  into another problem  $B$ .

**Reduction**  $A \leq_m B$ <sup>2</sup> :  $A$  is **reducible** to  $B$  if solutions to  $B$  exist and whenever given the solutions  $A$  can be solved.

Solving  $A$  **cannot be harder** than solving  $B$ .

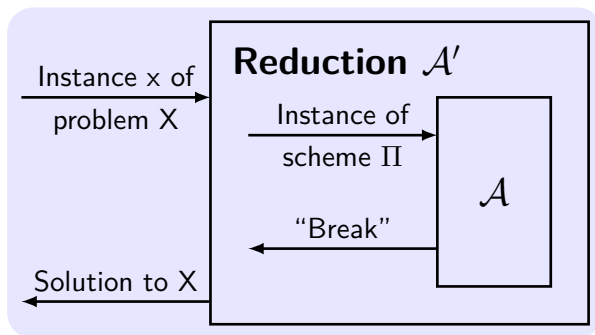
## Example

- “measure the area of a rectangle”  $\leq_m$  “measure the length and width of rectangle”
- “calculate  $x^2$ ”  $\leq_m$  “calculate  $x \times y$ ”

---

<sup>2</sup> $_m$  means the mapping reduction.

# Proofs of Reduction

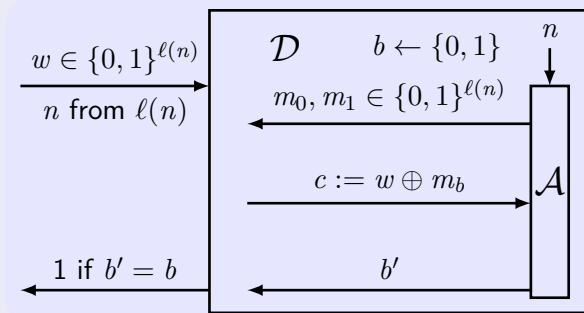


- A PPT  $\mathcal{A}$  can break  $\Pi$  with probability  $\varepsilon(n)$ .
- **Assumption:** Problem  $X$  is *hard* to solve.
- **Reduction:** Reduce  $\mathcal{A}'$  to  $\mathcal{A}$ .  $\mathcal{A}'$  solves  $x$  efficiently with probability  $1/p(n)$ , running  $\mathcal{A}$  as a sub-routine.
- **Contradiction:** If  $\varepsilon(n)$  is non-negligible, then  $\mathcal{A}'$  solves  $X$  efficiently with non-negligible probability  $\varepsilon(n)/p(n)$ .

# Proof of Indistinguishable Encryptions

**Idea:** Use  $\mathcal{A}$  to construct  $D$  for  $G$ , so that  $D$  distinguishes  $G$  when  $\mathcal{A}$  breaks  $\tilde{\Pi}$ . Since  $D$  cannot distinguish  $G$ , so that  $\mathcal{A}$  cannot break  $\tilde{\Pi}$ .

## Proof.



$$\Pr[D(w) = 1] = \Pr[\text{PrivK}_{\mathcal{A}, \tilde{\Pi}}^{\text{eav}}(n) = 1]$$



# Proof of Indistinguishable Encryptions (Cont.)

## Proof.

To prove  $\varepsilon(n) \stackrel{\text{def}}{=} \Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] - \frac{1}{2}$  is negligible.

(1) If  $w$  is  $r$  chosen *u.a.r.*, then  $\tilde{\Pi}$  is OTP.

$$\Pr[D(r) = 1] = \Pr[\text{PrivK}_{\mathcal{A}, \tilde{\Pi}}^{\text{eav}}(n) = 1] = \frac{1}{2};$$

(2) If  $w$  is  $G(k)$ , then  $\tilde{\Pi} = \Pi$ .

$$\Pr[D(G(k)) = 1] = \Pr[\text{PrivK}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] = \frac{1}{2} + \varepsilon(n).$$

Use Definition 4:

$$|\Pr[D(r) = 1] - \Pr[D(G(k)) = 1]| = \varepsilon(n) \leq \text{negl}(n).$$



# Handling Variable-Length Messages (homework)

## Definition 7

A **deterministic** polynomial-time algorithm  $G$  is a **variable output-length pseudorandom generator** if

- 1  $G(s, 1^\ell)$  outputs a string of length  $\ell > 0$ , where  $s$  is a string.
- 2  $G(s, 1^\ell)$  is a prefix of  $G(s, 1^{\ell'})$ ,  $\ell' > \ell$ .<sup>3</sup>
- 3  $G_\ell(s) \stackrel{\text{def}}{=} G(s, 1^{\ell(|s|)})$ . Then  $\forall \ell(\cdot)$ ,  $G_\ell$  is a PRG with expansion factor  $\ell$ .

Both Construction 5 and Theorem 6 hold here.

---

<sup>3</sup>for technical reasons to prove security.

# Computational Security vs. Info.-theoretical Security

	<b>Computational</b>	<b>Info.-theoretical</b>
<b>Adversary</b>	PPT eavesdropping	no limited eavesdropping
<b>Definition</b>	indistinguishable $\frac{1}{2} + \text{negl}$	indistinguishable $\frac{1}{2}$
<b>Assumption</b>	pseudorandom	random
<b>Key</b>	short random str.	long random str.
<b>Construction</b>	XOR pad	XOR pad
<b>Prove</b>	reduction	prob. theory