

Private-Key Encryption and Pseudorandomness (Part I)

Yu Zhang

HIT/CST/NIS

Cryptography, Spring, 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

Computational security vs. Information-theoretical security

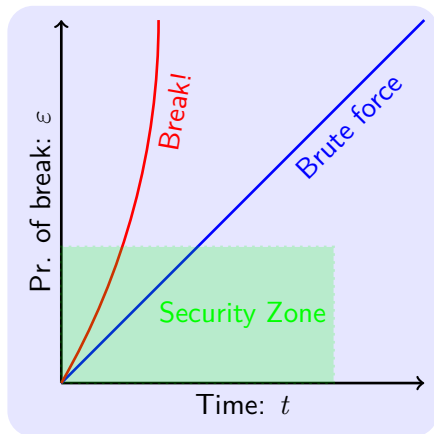
Kerckhoffs's Another Principle

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

- Information-theoretical security: 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. $1/|\mathcal{K}|$).



Concrete Approach

A scheme is (t, ε) -**secure** if every adversary running for time at most t succeeds in breaking the scheme with probability at most ε .

Example

Optimal security: when the key has length n , an adversary running in time t can succeed with probability at most $t/2^n$.

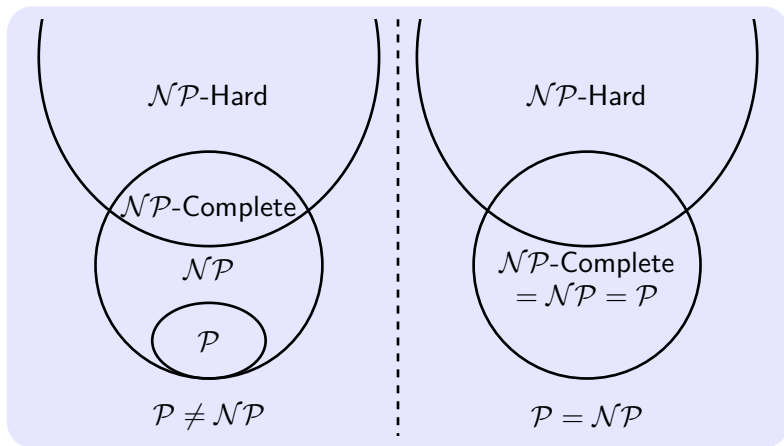
$t = 2^{80}$ 2^{20} 1GHz CPUs run 35 years.

$n = 128$ 2^{170} atoms in the planet.

$\varepsilon = 2^{-48}$ once every 100 years with probability $2^{-30}/\text{sec}$.

- But one may ask: What type of computing power? What if under Moore's Law? Which kind of implementation? What is the success probability of running for 10 years? Does the key length matter?

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



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

This is very dangerous!

Asymptotic Approach

Time t and probability ε are described as functions of **security parameter** n (usually, the length of key).

- Time is PPT (**probabilistic polynomial-time**): n^a for constant a .
- Probability is **negligible**: smaller than any inverse polynomial n^{-b} for constant b .
- **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} .

- 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.
- A **probabilistic** algorithm has the capability of “tossing coins”.
- A can run another PPT A' as a sub-routine in polynomial-time.
- 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 $f(n) < \frac{1}{p(n)}$.
- $\text{negl}_3(n) = \text{negl}_1(n) + \text{negl}_2(n)$ is negligible.
- positive polynomial p , $\text{negl}_4(n) = p(n) \cdot \text{negl}_1(n)$ is negligible.
- Problem X is *hard* if X cannot be solved by any polynomial-time algorithm except with negligible probability.

Remarks on The Asymptotic Approach

- The longer the key, the higher the security.
- Increasing n to defend against increases in computing power.
- Convention: algorithm input is 1^n . (a string of n 1's)

Example

	Honest party	Adversary	
	$10^6 \cdot n^2$	$10^8 \cdot n^4$	2^{20-n}
1GHz $n = 50$	2.5 sec.	1 week	2^{-30}
16GHz $n = 500$	0.625 sec.	16 week	2^{-80}

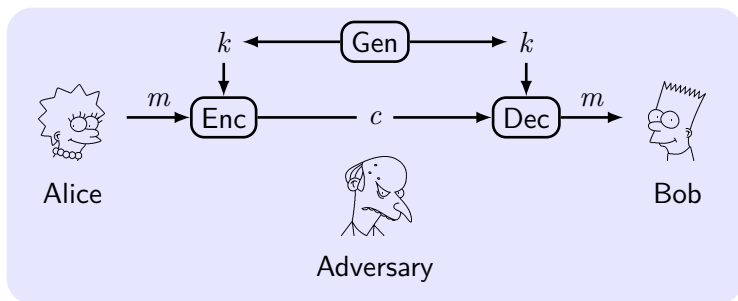
1 A Computational Approach to Cryptography

2 Defining Computationally-Secure Encryption

3 Pseudorandomness

4 Constructing Secure Encryption Schemes

Defining Private-key Encryption Scheme



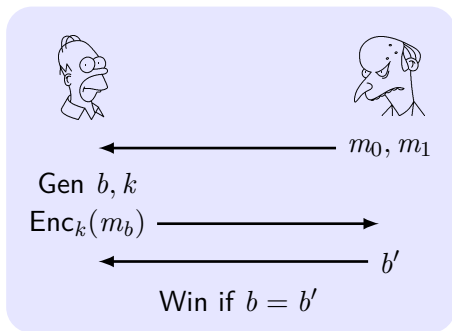
A **Private-key encryption scheme** Π is a tuple of PPT $(\text{Gen}, \text{Enc}, \text{Dec})$

- $k \leftarrow \text{Gen}(1^n)$, $|k| \geq n$ (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.



Definition 1

Π has **indistinguishable encryptions in the presence of an eavesdropper** if \forall PPT \mathcal{A} , \exists a negligible function 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} .

Properties of Definition of Indistinguishable

- 1 No single bit** can be guessed in a randomly-chosen plaintext with probability significantly better than $1/2$.

$$\Pr[\mathcal{A}(1^n, \text{Enc}_k(m)) = m^i] \leq \frac{1}{2} + \text{negl}(n).$$

where m^i is the i th bit of m , $\mathcal{A}(\cdot)$ outputs the guess.

- 2 No function of plaintext** can be learned regardless of the *a priori* distribution over the plaintext.

$\forall \mathcal{A}, \exists \mathcal{A}', \forall f$ and $\forall S \in \{0, 1\}^*$,

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

where $m \in S_n \stackrel{\text{def}}{=} S \cap \{0, 1\}^n$.

Intuition: No partial information leaks.

Definition 2

Π 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.
- It is impossible to definitively prove randomness.



Distinguisher: Statistical Tests

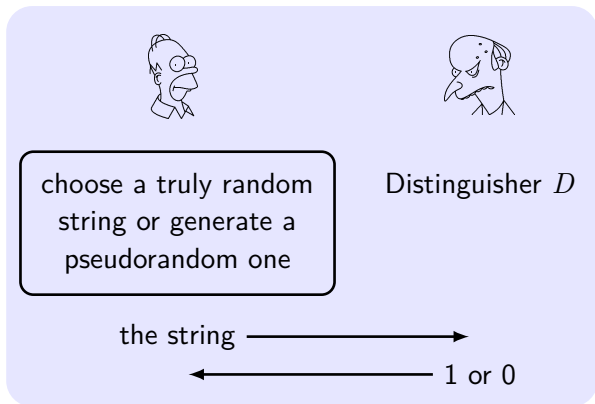
The pragmatic approach is to take many sequences of random numbers from a given generator and subject them to a battery of statistical tests.¹

- $D(x) = 0$ 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 $\text{max-run-of-0}(x) \leq 10 \cdot \log n$

¹State-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 .

Remarks on Pseudorandom Generators

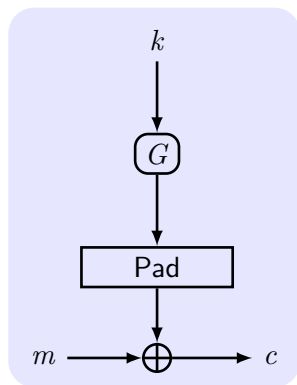
- **Sparse outputs:** In the case of $\ell(n) = 2n$, only 2^{-n} of strings of length $2n$ occurs.
- **Brute force attack:** Given an unlimited amount of time, one can distinguish $G(s)$ from r with a high probability by generating all strings with all seeds.

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

- **Sufficient seed space:** s must be long enough against brute force attack.
- **Existence:** Under the weak assumption that *one-way functions* exists, or $\mathcal{P} \neq \mathcal{NP}$.

- 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 $A \leq_m B$ ² : A is **reducible** to B if solutions to B exist and whenever given the solutions A can be solved.

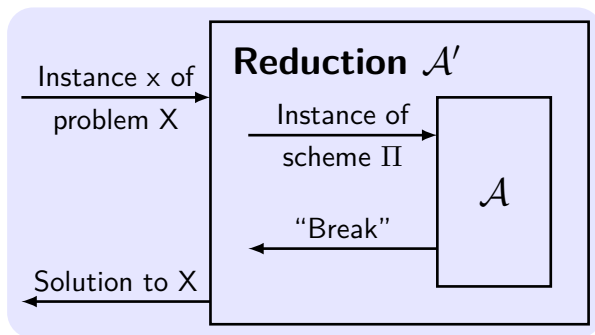
Solving A **cannot be harder** than solving B .

Example

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

² $_m$ means the mapping reduction.

Proofs of Reduction

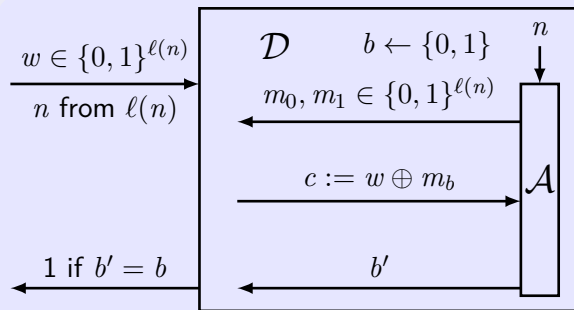


- A PPT \mathcal{A} can break Π with probability $\varepsilon(n)$.
- **Assumption:** Problem X is *hard* to solve.
- **Reduction:** Reduce \mathcal{A}' to \mathcal{A} . \mathcal{A}' solves x efficiently with probability $1/p(n)$, running \mathcal{A} as a sub-routine.
- **Contradiction:** If $\varepsilon(n)$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

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



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 $G(s, 1^{\ell'})$, $\ell' > \ell$.³
- 3 $G_\ell(s) \stackrel{\text{def}}{=} G(s, 1^{\ell(|s|)})$. Then $\forall \ell(\cdot)$, G_ℓ is a PRG with expansion factor ℓ .

Both Construction 5 and Theorem 6 hold here.

³for technical reasons to prove security.

Computational Security vs. Info.-theoretical Security

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