Department of
Computer Science and Engineering
计算机科学与工程系

CSE5014 CRYPTOGRAPHY AND NETWORK SECURITY

Dr. Qi Wang

Department of Computer Science and Engineering
Office: Room413, CoE South Tower
Email: wangqi@sustech.edu.cn

Perfect secrecy

- **Definition 1.5** An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if for every distribution over $\mathcal{M}$, every $m \in \mathcal{M}$, and every $c \in \mathcal{C}$ with $\Pr[C = c] > 0$, it holds that

$$\Pr[M = m | C = c] = \Pr[M = m]$$

Perfect secrecy

- **Definition 1.5** An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if for every distribution over $\mathcal{M}$, every $m \in \mathcal{M}$, and every $c \in \mathcal{C}$ with $\Pr[C = c] > 0$, it holds that

$$\Pr[M = m | C = c] = \Pr[M = m]$$

Equivalently, for every set $M \subseteq \{0,1\}^{\ell}$ of plaintexts, and for every strategy used by Eve, if we choose at random $x \in M$ and a random key $k \in \{0,1\}^{n}$, then the probability that Eve guesses x after seeing $Enc_k(x)$ is at most $1/|M|$, i.e.,

$$\Pr[Eve(Enc_k(x)) = x] \leq 1/|M|$$

Perfect secrecy

- Another two equivalent definitions

Perfect secrecy

- Another two equivalent definitions

  Definition 1.6 *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{x_0, x_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses x_b after seeing the ciphertext $c = Enc_k(x_b)$ is at most $1/2$.

Perfect secrecy

- Another two equivalent definitions

Definition 1.6 *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{x_0, x_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0,1\}$ and a random key $k \in \{0,1\}^n$, then the probability that Eve guesses x_b after seeing the ciphertext $c = Enc_k(x_b)$ is at most $1/2$.

Definition 1.7 *Perfect secrecy*. Two probability distributions X, Y over $\{0,1\}^\ell$ are *identical*, denoted by $X \equiv Y$, if for every $y \in \{0,1\}^\ell$, $\Pr[X = y] = \Pr[Y = y]$. An encryption scheme (Gen, Enc, Dec) is *perfectly secure* if for every pair of plaintexts x, x', we have $Enc_{U_n}(x) \equiv Enc_{U_n}(x')$.

Perfect security

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{x_0, x_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses x_b after seeing the ciphertext $c = Enc_k(x_b)$ is at most $1/2$.

  Key point: The ciphertext c reveals zero additional information about the plaintext m.

$$\Pr[M = m|C = c] = Pr[M = m]$$

Perfect security

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{x_0, x_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0,1\}$ and a random key $k \in \{0,1\}^n$, then the probability that Eve guesses x_b after seeing the ciphertext $c = Enc_k(x_b)$ is at most $1/2$.

  Key point: The ciphertext c reveals zero additional information about the plaintext m.

  $$\Pr[M = m|C = c] = Pr[M = m]$$

  Equivalently, for every set $M \subseteq \{0,1\}^\ell$ of plaintexts, and for every strategy used by Eve, if we choose at random $x \in M$ and a random key $k \in \{0,1\}^n$, then the probability that Eve guesses x after seeing $Enc_k(x)$ is at most $1/|M|$, i.e.,

  $$\Pr[Eve(Enc_k(x)) = x] \leq 1/|M|$$

Perfect Secrecy

- **Theorem 1.8** (Two-to-Many Theorem) The scheme (Gen, Enc, Dec) is *perfectly secure* if and only if

$$Pr[Eve(Enc_k(x_b)) = x_b] \leq 1/2.$$

Perfect Secrecy

- **Theorem 1.8** (Two-to-Many Theorem) The scheme (Gen, Enc, Dec) is *perfectly secure* if and only if

  $$Pr[Eve(Enc_k(x_b)) = x_b] \leq 1/2.$$

  Proof.

Perfect Secrecy

- **Theorem 1.8** (Two-to-Many Theorem) The scheme (Gen, Enc, Dec) is *perfectly secure* if and only if
  $$Pr[Eve(Enc_k(x_b)) = x_b] \leq 1/2.$$

  Proof.
  The "only if" part is easy (by definition, this is the special case that $|M| = 2$).

  The "if" part is tricky.

Perfect Secrecy

- **Theorem 1.8** (Two-to-Many Theorem) The scheme (Gen, Enc, Dec) is *perfectly secure* if and only if

$$Pr[Eve(Enc_k(x_b)) = x_b] \leq 1/2.$$

Proof.

The "only if" part is easy (by definition, this is the special case that $|M| = 2$).

The "if" part is tricky.

We need to show that if there is some set M and some strategy for Eve to guess a plaintext chosen from M with probability larger than $1/|M|$, then there is also some set M' of size 2 and a strategy Eve' for Eve to guess a plaintext chosen from M' with probability larger than $1/2$.

We fix $x_0 = 0^\ell$ and pick x_1 at random in M. Then it holds that for random key k and message $x_1 \in M$,

$$\Pr_{k \leftarrow \{0,1\}^n, x_1 \leftarrow M}[Eve(Enc_k(x_1)) = x_1] > 1/|M|.$$

On the other hand, for every choice of k, $x' = Eve(Enc_k(x_0))$ is a fixed string independent on the choice of x_1, and so if we pick x_1 at random in M, then the probability that $x_1 = x'$ is at most $1/|M|$, i.e.,

$$\Pr_{k \leftarrow \{0,1\}^n, x_1 \leftarrow M}[Eve(Enc_k(x_0)) = x_1] \leq 1/|M|.$$

Due to the linearity of expection,there exists some x_1 satisfying

$$\Pr[Eve(Enc_k(x_1)) = x_1] > \Pr[Eve(Enc_k(x_0)) = x_1]. \text{ (why?)}$$

We now define a new attacker Eve' as: $Eve'(c) = \begin{cases} x_1, & \text{if } Eve(c) = x_1 \\ x_i, i \in \{0, 1\} \text{ at random}, & \text{otherwise} \end{cases}$

This means the probability that $Eve'(Enc_k(x_b)) = x_b$ is larger than $1/2$ (Why?).

One-time Pad

- The *XOR* operation: $a \oplus b = a + b \bmod 2$.

One-time Pad

- The *XOR* operation: $a \oplus b = a + b \bmod 2$.

$$a \oplus 0 = a$$
$$a \oplus a = 0$$
$$a \oplus b = b \oplus a \text{ (Commutativity)}$$
$$a \oplus (b \oplus c) = (a \oplus b) \oplus c \text{ (Associativity)}$$

One-time Pad

- The *XOR* operation: $a \oplus b = a + b \bmod 2$.

$$a \oplus 0 = a$$
$$a \oplus a = 0$$
$$a \oplus b = b \oplus a \text{ (Commutativity)}$$
$$a \oplus (b \oplus c) = (a \oplus b) \oplus c \text{ (Associativity)}$$

The One-time Pad scheme (Vernam 1917, Shannon 1949):
$n = |k| = |x|$, $Enc : \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$
$Enc_k(x) = x \oplus k$
$Dec_k(y) = y \oplus k$

One-time Pad

- The *XOR* operation: $a \oplus b = a + b \bmod 2$.

$$a \oplus 0 = a$$
$$a \oplus a = 0$$
$$a \oplus b = b \oplus a \text{ (Commutativity)}$$
$$a \oplus (b \oplus c) = (a \oplus b) \oplus c \text{ (Associativity)}$$

The One-time Pad scheme (Vernam 1917, Shannon 1949):

$n = |k| = |x|$, $Enc : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$

$Enc_k(x) = x \oplus k$

$Dec_k(y) = y \oplus k$

Validity:

$$Dec_k(Enc_k(x)) = (x \oplus k) \oplus k = x \oplus (k \oplus k) = x \oplus 0^n = x$$

One-time Pad

- **Theorem 1.9** One-time pad is *perfectly secure*.

One-time Pad

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Proof. Prove that for every $x \in \{0,1\}^n$, the distribution $Y_x = Enc_{U_n}(x)$ is uniformly distributed.

One-time Pad

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Proof. Prove that for every $x \in \{0,1\}^n$, the distribution $Y_x = Enc_{U_n}(x)$ is uniformly distributed.

  Let $y \in \{0,1\}^n$, we need to show that
  $Pr_{k \leftarrow_R \{0,1\}^n}[x \oplus k = y] = 2^{-n}$
  Since there is a unique single value of $k = x \oplus y$, the probability that the equation is true is 2^{-n}.

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Proof. Fix arbitrary distributuion over $\mathcal{M} = \{0,1\}^n$, and arbitrary $m, c \in \{0,1\}^n$.

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Proof. Fix arbitrary distributuion over $\mathcal{M} = \{0,1\}^n$, and arbitrary $m, c \in \{0,1\}^n$.

  $\Pr[M = m \mid C = c] =?$

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Proof. Fix arbitrary distributuion over $\mathcal{M} = \{0,1\}^n$, and arbitrary $m, c \in \{0,1\}^n$.

  $\Pr[M = m \mid C = c] =?$
  $= \Pr[C = c \mid M = m] \cdot \Pr[M = m] / \Pr[C = c]$

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

Proof. Fix arbitrary distributuion over $\mathcal{M} = \{0,1\}^n$, and arbitrary $m, c \in \{0,1\}^n$.

$$\Pr[M = m \mid C = c] = ?$$
$$= \Pr[C = c \mid M = m] \cdot \Pr[M = m] / \Pr[C = c]$$

$$\Pr[C = c]$$
$$= \sum_{m'} \Pr[C = c \mid M = m'] \cdot \Pr[M = m']$$
$$= \sum_{m'} \Pr[K = m' \oplus c] \cdot \Pr[M = m']$$
$$= \sum_{m'} 2^{-n} \cdot \Pr[M = m']$$
$$= 2^{-n}$$

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

Proof. Fix arbitrary distributuion over $\mathcal{M} = \{0,1\}^n$, and arbitrary $m, c \in \{0,1\}^n$.

$$
\begin{aligned}
&\Pr[M = m \mid C = c] = ? \\
&= \Pr[C = c \mid M = m] \cdot \Pr[M = m] / \Pr[C = c] \\
&= \Pr[K = m \oplus c] \cdot \Pr[M = m] / 2^{-n} \\
&= 2^{-n} \cdot \Pr[M = m] / 2^{-n} \\
&= \Pr[M = m]
\end{aligned}
$$

$$
\begin{aligned}
&\Pr[C = c] \\
&= \sum_{m'} \Pr[C = c \mid M = m'] \cdot \Pr[M = m'] \\
&= \sum_{m'} \Pr[K = m' \oplus c] \cdot \Pr[M = m'] \\
&= \sum_{m'} 2^{-n} \cdot \Pr[M = m'] \\
&= 2^{-n}
\end{aligned}
$$

One-time Pad

- $\mathcal{Q}$: Is this the end of cryptography?

One-time Pad

- $\mathcal{Q}$: Is this the end of cryptography?

  We need more:

  ◇ Use the same key for many plaintexts

  ◇ Use n-bit key for $2n$-bit plaintexts.

One-time Pad

- $\mathcal{Q}$: Is this the end of cryptography?

  We need more:

  ◇ Use the same key for many plaintexts

  ◇ Use n-bit key for $2n$-bit plaintexts.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

One-time Pad

- $\mathcal{Q}$: Is this the end of cryptography?

  We need more:

  ⋄ Use the same key for many plaintexts

  ⋄ Use n-bit key for $2n$-bit plaintexts.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  Proof. See blackboard.

One-time Pad

- $\mathcal{Q}$: Is this the end of cryptography?

We need more:

◇ Use the same key for many plaintexts

◇ Use n-bit key for $2n$-bit plaintexts.

Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

Proof. See blackboard.

Suppose that (Gen, Enc, Dec) is such an encryption scheme. Denote by Y_0 the distribution $Enc_{U_{n-1}}(0^n)$ and by S_0 its support. Since there are only 2^{n-1} possible keys, we have $|S_0| \leq 2^{n-1}$.

Now for every key k the function $Enc_k(\cdot)$ is one-to-one and hence its image is of size $\geq 2^n$. This means that for every k, there exists x such that $Enc_k(x) \notin S_0$. Fix such a k and x, then the distribution $Enc_{U_{n-1}}(x)$ does not have the same support as Y_0 and hence it is not identical to Y_0.

Key generation

- When describing algorithms, we assume access to uniformly distributed bits/bytes. Where do these actually come from?

Key generation

- When describing algorithms, we assume access to uniformly distributed bits/bytes. Where do these actually come from?

- Precise details depend on the system
  - Linux or unix: /dev/random or /dev/urandom
  - Do not use *rand()* or *java.util.Random*
  - Use crypto libraries instead

Random-number generation

- Two steps:
  1. Continually collect a “pool” of high-entropy (“unpredictable”) data
  2. (Smoothing) When random bits are requested, process this data to generate a sequence of uniform, independent bits/bytes

Random-number generation

- Two steps:

  1. Continually collect a “pool” of high-entropy (“unpredictable”) data
     - Must ultimately come from some physical process (keystroke/mouse movements, delays between network events, hard-disk access times, hardware random-number generation (Intel), etc.)

  2. (Smoothing) When random bits are requested, process this data to generate a sequence of uniform, independent bits/bytes

Random-number generation

- Two steps:
  1. Continually collect a “pool” of high-entropy (“unpredictable”) data
     - Must ultimately come from some physical process (keystroke/mouse movements, delays between network events, hard-disk access times, hardware random-number generation (Intel), etc.)
  2. (Smoothing) When random bits are requested, process this data to generate a sequence of uniform, independent bits/bytes
     - Need to eliminate both *bias* and *dependencies*

Random-number generation

- Step 2: Smoothing

  von Neumann technique for eliminating *bias*:

  - Collect two bits per output bit
    - $01 \rightarrow 0,\ 10 \rightarrow 1,\ 00, 11 \rightarrow$ skip
  - Note that this assumes *independence* (as well as constant bias)

Random-number generation

- Step 2: Smoothing

  von Neumann technique for eliminating *bias*:

  - Collect two bits per output bit
    - $01 \rightarrow 0$, $10 \rightarrow 1$, $00, 11 \rightarrow$ skip
  - Note that this assumes *independence* (as well as constant bias)

- Read desired number of bytes from “/dev/urandom”

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  - The key is as long as the message
  - Only secure if each key is used to encrypt a single message
  - Trivially broken by a known-plaintext attack

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  - The key is as long as the message
  - Only secure if each key is used to encrypt a single message
  - Trivially broken by a known-plaintext attack

  $\mathcal{Q}$: What about if using the same key twice?

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  – The key is as long as the message

  – Only secure if each key is used to encrypt a single message

  – Trivially broken by a known-plaintext attack

  $\mathcal{Q}$: What about if using the same key twice?

  Say $c_1 = k \oplus m_1$ and $c_2 = k \oplus m_2$.

  $$c_1 \oplus c_2 = (k \oplus m_1) \oplus (k \oplus m_2) = m_1 \oplus m_2$$

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  - The key is as long as the message
  - Only secure if each key is used to encrypt a single message
  - Trivially broken by a known-plaintext attack

  $\mathcal{Q}$: What about if using the same key twice?

  Say $c_1 = k \oplus m_1$ and $c_2 = k \oplus m_2$.

  $c_1 \oplus c_2 = (k \oplus m_1) \oplus (k \oplus m_2) = m_1 \oplus m_2$

  $m_1 \oplus m_2$ leaks information about m_1, m_2

Perfect security

- **Theorem 1.9** One-time pad is *perfectly secure*.

  Theorem 1.10 (Limitations of perfect secrecy) There is no *perfectly secure* encryption schemes (Gen, Enc, Dec) with n-bit plaintexts and $(n-1)$-bit keys.

  - The key is as long as the message
  - Only secure if each key is used to encrypt a single message
  - Trivially broken by a known-plaintext attack

  Whenever faced with an impossibility result, it is a good idea to examine whether we can relax these assumptions to still get what we want (or at least something close to that).

Statistical Security

- Eve has a tiny advantage in its posteriori guessing probability compared to the priori probability.

Statistical Security

- Eve has a tiny advantage in its posteriori guessing probability compared to the priori probability.

  We may say that an encryption scheme is *ϵ-statistically indistinguishable* if the probability that Eve guesses which of the two messages was encrypted is at most $1/2 + \epsilon$.

Statistical Security

- Eve has a tiny advantage in its posteriori guessing probability compared to the priori probability.

  We may say that an encryption scheme is *ϵ-statistically indistinguishable* if the probability that Eve guesses which of the two messages was encrypted is at most $1/2 + \epsilon$.

  $\mathcal{Q}$:
  1) Is the relaxed definition still strong enough in practice?
  2) Does the relaxation buy something we could not get with the original definition (perfect security)?

Statistical Security

- Eve has a tiny advantage in its posteriori guessing probability compared to the priori probability.

  We may say that an encryption scheme is *ϵ-statistically indistinguishable* if the probability that Eve guesses which of the two messages was encrypted is at most $1/2 + \epsilon$.

  $\mathcal{Q}$:
  1) Is the relaxed definition still strong enough in practice?
  2) Does the relaxation buy something we could not get with the original definition (perfect security)?

  $\mathcal{A}$:
  1) Yes, if ϵ is small (say 10^{-6} or even 10^{-100})
  2) No, we cannot have key shorter than the message.

ϵ-Statistical Security

- **Definition 2.1** Let X and Y be two distributions over $\{0,1\}^n$. The *statistical distance* of X and Y, denoted by $\Delta(X, Y)$ is defined to be
  $$\max_{T \subseteq \{0,1\}^n} |\Pr[X \in T] - \Pr[Y \in T]|.$$
  If $\Delta(X, Y) \leq \epsilon$, we say that $X \equiv_\epsilon Y$.

ϵ-Statistical Security

- **Definition 2.1** Let X and Y be two distributions over $\{0,1\}^n$. The *statistical distance* of X and Y, denoted by $\Delta(X,Y)$ is defined to be

  $$\max_{T\subseteq\{0,1\}^n} |\Pr[X\in T]-\Pr[Y\in T]|.$$

  If $\Delta(X,Y)\leq\epsilon$, we say that $X\equiv_\epsilon Y$.

  Definition 2.2 *ϵ-Statistical Security*. An encryption scheme (Gen, Enc, Dec) is *ϵ-statistically secure* if for every pair of plaintexts m, m', we have $Enc_{U_n}(m)\equiv_\epsilon Enc_{U_n}(m')$.

ϵ-Statistical Security

- **Lemma 2.3**

$$\Delta(X, Y) = \frac{1}{2} \sum_{w \in Supp(X) \cup Supp(Y)} |Pr[X = w] - Pr[Y = w]|$$

ϵ-Statistical Security

- **Lemma 2.3**

$$\Delta(X, Y) = \frac{1}{2} \sum_{w \in Supp(X) \cup Supp(Y)} |Pr[X = w] - Pr[Y = w]|$$

Proof. See blackboard.

ϵ-Statistical Security

- **Lemma 2.3**

$$\Delta(X,Y) = \frac{1}{2} \sum_{w \in Supp(X) \cup Supp(Y)} |Pr[X = w] - Pr[Y = w]|$$

Proof. See blackboard.

Observations:

$$0 \leq \Delta(X,Y) \leq 1$$
$$\Delta(X,Y) = 0 \text{ if } X = Y$$
$$0 \leq \Delta(X,Y) \leq \Delta(X,Z) + \Delta(Z,Y)$$

ϵ-Statistical Security

- **Lemma 2.3**

$$\Delta(X, Y) = \frac{1}{2} \sum_{w \in Supp(X) \cup Supp(Y)} |Pr[X = w] - Pr[Y = w]|$$

Proof. See blackboard.

Observations:

$$0 \leq \Delta(X, Y) \leq 1$$
$$\Delta(X, Y) = 0 \text{ if } X = Y$$
$$0 \leq \Delta(X, Y) \leq \Delta(X, Z) + \Delta(Z, Y)$$

Δ is a *metric*.

ϵ-Statistical Security

- **Lemma 2.4** Eve has at most $1/2 + \epsilon$ success probability if and only if for every pair of m_1, m_2,

  $$\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) \leq 2\epsilon.$$

ϵ-Statistical Security

- **Lemma 2.4** Eve has at most $1/2 + \epsilon$ success probability if and only if for every pair of m_1, m_2,

$$\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) \leq 2\epsilon.$$

Proof.

Suppose that Eve has $1/2 + \epsilon$ success probability with m_1, m_2. Let $p_{i,j} = \Pr[Eve(Enc_{U_n}(m_i)) = j]$. Then we have

$$p_{1,1} + p_{1,2} = 1$$
$$p_{2,1} + p_{2,2} = 1$$
$$(1/2)p_{1,1} + (1/2)p_{2,2} \leq 1/2 + \epsilon.$$

The last two together imply that

$$p_{1,1} - p_{2,1} \leq 2\epsilon,$$

which means that if we let T be the set $\{c : Eve(c) = 1\}$, then T demonstrates that $\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) \leq 2\epsilon$.

Similarly, if we have such a set T, we can define an attacker from it that succeeds with probability $1/2 + \epsilon$.

Limitation of ϵ-Statistical Security

- **Theorem 2.5** Let (Gen, Enc, Dec) be a valid encryption with $Enc : \{0,1\}^{n} \times \{0,1\}^{n+1} \to \{0,1\}^*$. Then there exist plaintexts m_1, m_2 with $\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) > 1/2$.

Limitation of ϵ-Statistical Security

- **Theorem 2.5** Let (Gen, Enc, Dec) be a valid encryption with $Enc : \{0,1\}^{n} \times \{0,1\}^{n+1} \to \{0,1\}^*$. Then there exist plaintexts m_1, m_2 with $\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) > 1/2$.

Proof. See blackboard.

Fact. For a random variable Y, if $E[Y] \le \mu$ the $\Pr[Y \le \mu] > 0$.
Let $m_1 = 0^{n+1}$, and let $S = Supp(Enc_{U_n}(m_1))$, then $|S| \le 2^n$.
We choose a random message $m \leftarrow_R \{0,1\}^{n+1}$ and define the following 2^n random variables for every k:

$$T_k(m) = \begin{cases} 1, & \text{if } Enc_k(m) \in S \\ 0, & \text{otherwise} \end{cases}$$

Since for every k, $Enc_k(\cdot)$ is one-to-one, we have $\Pr[T_k = 1] \le 1/2$. Define $T = \sum_{k \in \{0,1\}^n} T_k$, then

$$E[T] = E[\sum_k T_k] = \sum_k E[T_k] \le 2^n/2.$$

This means the probability $\Pr[T \le 2^n/2] > 0$. In other words, there exists an m s.t. $\sum_k T_k(m) \le 2^n/2$. For such m, at most half of the keys k satisfy $Enc_k(m) \in S$, i.e.,

$$\Pr[Enc_{U_n}(m) \in S] \le 1/2.$$

Since $\Pr[Enc_{U_n}(0^{n+1}) \in S] = 1$, we have

$$\Delta(Enc_{U_n}(0^{n+1}), Enc_{U_n}(m)) > 1/2.$$

Limitation of ϵ-Statistical Security

- **Theorem 2.5** Let (Gen, Enc, Dec) be a valid encryption with $Enc : \{0,1\}^n \times \{0,1\}^{n+1} \to \{0,1\}^*$. Then there exist plaintexts m_1, m_2 with $\Delta(Enc_{U_n}(m_1), Enc_{U_n}(m_2)) > 1/2$.

Proof. See blackboard.

Fact. For a random variable Y, if $E[Y] \le \mu$ the $\Pr[Y \le \mu] > 0$.
Let $m_1 = 0^{n+1}$, and let $S = Supp(Enc_{U_n}(m_1))$, then $|S| \le 2^n$.
We choose a random message $m \leftarrow_R \{0,1\}^{n+1}$ and define the followin

$$T_k(m) = \begin{cases} 1, & \text{if } Enc_k(m) \in S \\ 0, & \text{otherwise} \end{cases}$$

Since for every k, $Enc_k(\cdot)$ is one-to-one, we have $\Pr[T_k = 1] \le 1/2$. D

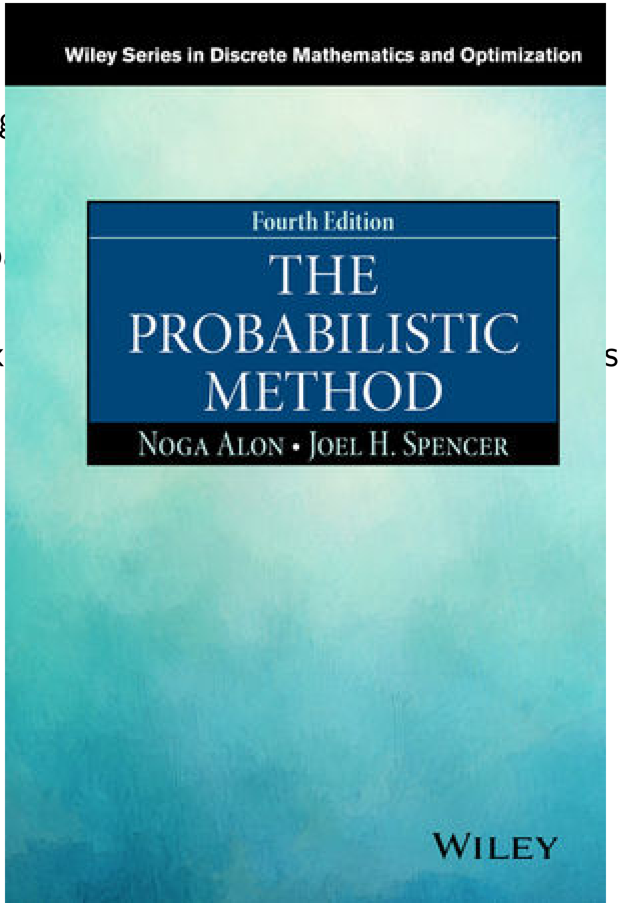
$$E[T] = E[\sum_k T_k] = \sum_k E[T_k] \le 2^n/2.$$

This means the probability $\Pr[T \le 2^n/2] > 0$. In other words, there ex such m, at most half of the keys k satisfy $Enc_k(m) \in S$, i.e.,

$$\Pr[Enc_{U_n}(m) \in S] \le 1/2.$$

Since $\Pr[Enc_{U_n}(0^{n+1}) \in S] = 1$, we have

$$\Delta(Enc_{U_n}(0^{n+1}), Enc_{U_n}(m)) > 1/2.$$



Wiley Series in Discrete Mathematics and Optimization

Fourth Edition

THE PROBABILISTIC METHOD

NOGA ALON • JOEL H. SPENCER

WILEY

Computational Security

- Statistical security does not allow us to break the impossibility result.

Computational Security

- Statistical security does not allow us to break the impossibility result.
  - In real life, people are using encryption with keys shorter than the message size to encrypt all kinds of sensitive information.
  - If the algorithm you use to break the encryption scheme runs in time 2^n , it seems OK since the message may be expired by then ...

Computational Security

- Statistical security does not allow us to break the impossibility result.
  - In real life, people are using encryption with keys shorter than the message size to encrypt all kinds of sensitive information.
  - If the algorithm you use to break the encryption scheme runs in time 2^n , it seems OK since the message may be expired by then ...
- **Idea**: Would be OK if a scheme leaked information with *tiny probability* to eavesdroppers with *bounded computational resources*

Computational Security

- Statistical security does not allow us to break the impossibility result.
  - In real life, people are using encryption with keys shorter than the message size to encrypt all kinds of sensitive information.
  - If the algorithm you use to break the encryption scheme runs in time 2^n , it seems OK since the message may be expired by then ...
- **Idea**: Would be OK if a scheme leaked information with *tiny probability* to eavesdroppers with *bounded computational resources*
  - Allowing security to “fail” with tiny probability
  - Restricting attention to “efficient” attackers

Tiny probability of failure?

- Say security fails with probability 2^{-60}

Tiny probability of failure?

- Say security fails with probability 2^{-60}
  - Should we be concerned about this?

Tiny probability of failure?

- Say security fails with probability 2^{-60}
  - Should we be concerned about this?
  - With probability $> 2^{-60}$, the sender and receiver will both be struck by lightning in the next year ...
  - Something that occurs with probability $2^{-60}/sec$ is expected to occur once every 100 billion years

Bounded attackers?

- Consider *brute-force search* of key space; assume one key can be tested per clock cycle

Bounded attackers?

- Consider *brute-force search* of key space; assume one key can be tested per clock cycle
  - Desktop computer $\approx 2^{57}$ keys/year
  - Supercomputer $\approx 2^{80}$ keys/year

Bounded attackers?

- Consider *brute-force search* of key space; assume one key can be tested per clock cycle
  - Desktop computer $\approx 2^{57}$ keys/year
  - Supercomputer $\approx 2^{80}$ keys/year
  - Supercomputer since Big Bang $\approx 2^{112}$ keys
    - Restricting attention to attackers who can try 2^{112} keys is fine!

Bounded attackers?

- Consider *brute-force search* of key space; assume one key can be tested per clock cycle
  - Desktop computer $\approx 2^{57}$ keys/year
  - Supercomputer $\approx 2^{80}$ keys/year
  - Supercomputer since Big Bang $\approx 2^{112}$ keys
    - Restricting attention to attackers who can try 2^{112} keys is fine!

Modern key space: 2^{128} keys or more ...

Computational Security

- Two problems:

Computational Security

- Two problems:

  1) While the particular algorithm runs in exponential time, we cannot guarantee that there is no other algorithm is efficient.

  2) We need a precise mathematical definition (like *prefect secrecy* definition).

Computational Security

- Two problems:

  1) While the particular algorithm runs in exponential time, we cannot guarantee that there is no other algorithm is efficient.

  e.g., the substitution cipher has a huge key space, but can be broken efficiently.

  2) We need a precise mathematical definition (like *prefect secrecy* definition).

Computational Security

- Two problems:

  1) While the particular algorithm runs in exponential time, we cannot guarantee that there is no other algorithm is efficient.

  e.g., the substitution cipher has a huge key space, but can be broken efficiently.

  2) We need a precise mathematical definition (like *prefect secrecy* definition).

  "Breaking E is very hard"?
  "Problem P cannot be solved in reasonable time"?

Computational Security

- Two problems:

1) While the particular algorithm runs in exponential time, we cannot guarantee that there is no other algorithm is efficient.

e.g., the substitution cipher has a huge key space, but can be broken efficiently.

2) We need a precise mathematical definition (like *prefect secrecy* definition).

"Breaking E is very hard"?
"Problem P cannot be solved in reasonable time"?

$\mathcal{Q}$: How do we model the resources of Eve (the adversary)?

Perfect indistinguishability

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{m_0, m_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses m_b after seeing the ciphertext $c = Enc_k(m_b)$ is at most $1/2$.

Perfect indistinguishability

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{m_0, m_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses m_b after seeing the ciphertext $c = Enc_k(m_b)$ is at most $1/2$.
- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

Perfect indistinguishability

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{m_0, m_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses m_b after seeing the ciphertext $c = Enc_k(m_b)$ is at most $1/2$.
- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

  Define a randomized experiment $PrivK_{A,\Pi}$:

  1. A outputs $m_0, m_1 \in \mathcal{M}$
  2. $k \leftarrow Gen$, $b \leftarrow \{0, 1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

Perfect indistinguishability

- **Definition 1.6** *Perfect secrecy*. An encryption scheme (Gen, Enc, Dec) with message space $\mathcal{M}$ and ciphertext space $\mathcal{C}$ is *perfectly secure* if and only if for every two distinct plaintexts $\{m_0, m_1\} \in \mathcal{M}$, and for every strategy used by Eve, if we choose at random $b \in \{0, 1\}$ and a random key $k \in \{0, 1\}^n$, then the probability that Eve guesses m_b after seeing the ciphertext $c = Enc_k(m_b)$ is at most $1/2$.
- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

  Define a randomized experiment $PrivK_{A,\Pi}$:

  1. A outputs $m_0, m_1 \in \mathcal{M}$
  2. $k \leftarrow Gen$, $b \leftarrow \{0, 1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

Perfect indistinguishability

- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

  Define a randomized experiment $PrivK_{A,\Pi}$:

  1. A outputs $m_0, m_1 \in \mathcal{M}$
  2. $k \leftarrow Gen$, $b \leftarrow \{0, 1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

Perfect indistinguishability

- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

  Define a randomized experiment $PrivK_{A,\Pi}$:

  1. A outputs $m_0, m_1 \in \mathcal{M}$
  2. $k \leftarrow Gen$, $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

  Π is *perfectly indistinguishable* if for all attackers (algorithms) A, it holds that $\Pr[PrivK_{A,\Pi} = 1] \leq 1/2$

Perfect indistinguishability

- Let $\Pi = (Gen, Enc, Dec)$ be an enryption scheme with message space $\mathcal{M}$, and A an adversary

  Define a randomized experiment $PrivK_{A,\Pi}$:

  1. A outputs $m_0, m_1 \in \mathcal{M}$
  2. $k \leftarrow Gen$, $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

  Π is *perfectly indistinguishable* if for all attackers (algorithms) A, it holds that $\Pr[PrivK_{A,\Pi} = 1] \leq 1/2$

  Claim: Π is *perfectly indistinguishable* $\Leftrightarrow$ Π is *perfectly secure*

Computational security?

- **Idea**: relax *perfect indistinguishability*

Computational security?

- **Idea**: relax *perfect indistinguishability*

Two approaches

- *Concrete* security
- *Asympototic* security

Computational indistinguishability

- (t, ϵ)-*indistinguishability* (concrete)
  - Security may fail with probability $\leq \epsilon$
  - Restrict attention to attackers running in time $\leq t$

Computational indistinguishability

- (t, ϵ)-*indistinguishability* (concrete)
  - Security may fail with probability $\leq \epsilon$
  - Restrict attention to attackers running in time $\leq t$

- Π is (t, ϵ)-indistinguishable if for all attackers A running in time at most t, it holds that

$$\Pr[PrivK_{A,\Pi} = 1] \leq 1/2 + \epsilon$$

Computational indistinguishability

- (t, ϵ)-*indistinguishability* (concrete)
  - Security may fail with probability $\leq \epsilon$
  - Restrict attention to attackers running in time $\leq t$

- Π is (t, ϵ)-indistinguishable if for all attackers A running in time at most t, it holds that

$$\Pr[PrivK_{A,\Pi} = 1] \leq 1/2 + \epsilon$$

Does not lead to a clean theory ...

- Sensitive to exact computational model
- Π can be (t, ϵ)-secure for many choices of t, ϵ

Computational indistinguishability

- Introduce *security parameter* n (asymptotic)
  - For now, can view as the key length
  - Fixed by honest parties at initialization
  - Known by adversary

Computational indistinguishability

- Introduce *security parameter* n (asymptotic)
  - For now, can view as the key length
  - Fixed by honest parties at initialization
  - Known by adversary

  Measure running time of all parties, and the success probability of the adversary, as functions of n

Computational indistinguishability

- Introduce *security parameter* n (asymptotic)
  - For now, can view as the key length
  - Fixed by honest parties at initialization
  - Known by adversary

Measure running time of all parties, and the success probability of the adversary, as functions of n

Computational indistinguishability:

- Security may fail with probability *negligible* in n
- Restrict attention to attackers running in time (at most) polynomial in n

Definitions

- A function $f : \mathbb{Z}^+ \to \mathbb{Z}^+$ is (at most) *polynomial* if there exists c s.t. $f(n) < n^c$ for large enough n.

  A function $f : \mathbb{Z}^+ \to [0,1]$ is *negligible* if every polynomial p it holds that $f(n) < 1/p(n)$ for large enough n.

  – Typical example: $f(n) = poly(n) \cdot 2^{-cn}$

Why these choices?

- "Efficient" $=$ "(probabilistic) polynomial-time (PPT)" borrowed from *complexity theory*

Why these choices?

- "Efficient" = "(probabilistic) polynomial-time (PPT)" borrowed from *complexity theory*

- Convenient closure properties
  - $poly * poly = poly$
    - Poly-many calls to PPT subroutine (with poly-size input) is still PPT
  - $poly * negl = negl$
    - Poly-many calls to subroutine that fails with negligible probability fails with negligible probability overall

(Re)defining encryption

- A *private-key encryption scheme* is defined by three PPT algorithms (Gen, Enc, Dec):
  - Gen: takes as input 1^n; outputs k
  - Enc: takes as input a key k and message $m \in \{0,1\}^*$; outputs ciphertext c: $c \leftarrow Enc_k(m)$
  - Dec: takes key k and ciphertext c as input; outputs a message m or "error" $(\perp)$

Computational indistinguishability (asymptotic)

- Fix Π, A
  Define a randomized experiment $PrivK_{A,\Pi}(n)$:
  1. $A(1^n)$ outputs $m_0, m_1 \in \{0,1\}^*$ of equal length
  2. $k \leftarrow Gen(1^n)$, $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

Computational indistinguishability (asymptotic)

- Fix Π, A

  Define a randomized experiment $PrivK_{A,\Pi}(n)$:

  1. $A(1^n)$ outputs $m_0, m_1 \in \{0,1\}^*$ of equal length
  2. $k \leftarrow Gen(1^n)$, $b \leftarrow \{0,1\}$, $c \leftarrow Enc_k(m_b)$
  3. $b' \leftarrow A(c)$

  Adversary A *succeeds* if $b = b'$, and we say the experiment evaluates to 1 in this case.

  Π is *computationally indistinguishable* (aka *EAV-secure*) if for all PPT attackers (algorithms) A, there is a *negligible* function ϵ such that

  $$\Pr[PrivK_{A,\Pi}(n) = 1] \leq 1/2 + \epsilon(n)$$

Example

- Consider a scheme where the best attack is *brute-force search* over the key space, and $Gen(1^n)$ generates a uniform n-bit key
  - So if A runs in time $t(n)$, then

    $$\Pr[PrivK_{A,\Pi}(n) = 1] < 1/2 + O(t(n)/2^n)$$

Example

- Consider a scheme where the best attack is *brute-force search* over the key space, and $Gen(1^n)$ generates a uniform n-bit key
  - So if A runs in time $t(n)$, then
    $$\Pr[PrivK_{A,\Pi}(n) = 1] < 1/2 + O(t(n)/2^n)$$
  - The scheme is EAV-secure: for any polynomial t, the function $t(n)/2^n$ is negligible.

Example

- Consider a scheme and a particular attacker A that runs for n^3 minutes and breaks the scheme with probability $2^{40}2^{-n}$
  - This does not contradict asymptotic security

Example

- Consider a scheme and a particular attacker A that runs for n^3 minutes and breaks the scheme with probability $2^{40}2^{-n}$
  - This does not contradict asymptotic security
  - What about real-world security (against this attacker)?
    - $n = 40$: A breaks with prob. 1 in 6 weeks
    - $n = 50$: A breaks with prob. $1/1000$ in 3 months
    - $n = 500$: A breaks with prob. 2^{-500} in 200 years

Example

- What happens when computers get faster?

  – Consider a scheme that takes time n^2 to run but time 2^n to break with prob. 1

Example

- What happens when computers get faster?

  – Consider a scheme that takes time n^2 to run but time 2^n to break with prob. 1

  What if computers get $4\times$ faster?

  – Users double n; maintain same running time

  – Attacker's work is (roughly) squared!

Encryption and plaintext length

- In practice, we want encryption schemes that can encrypt arbitrary-length messages.

Encryption and plaintext length

- In practice, we want encryption schemes that can encrypt arbitrary-length messages.

- In general, encryption does not hide the plaintext length
  - The definition takes this into account by requiring m_0, m_1 to have the same length.

Encryption and plaintext length

- In practice, we want encryption schemes that can encrypt arbitrary-length messages.

- In general, encryption does not hide the plaintext length
  - The definition takes this into account by requiring m_0, m_1 to have the same length.

- But leaking plaintext length can often lead to problems in the real world!
  - Databases searches
  - Encrypting compressed data

Micali & Goldwasser

Silvio Micali

Shafi Goldwasser

1984: semantic security, indistinguishability
(Turing Award 2012)

Micali & Blum

Silvio Micali

Manuel Blum

1984: defined notion of pseudo-random generator (Turing Award 1995)

Proof of Reduction

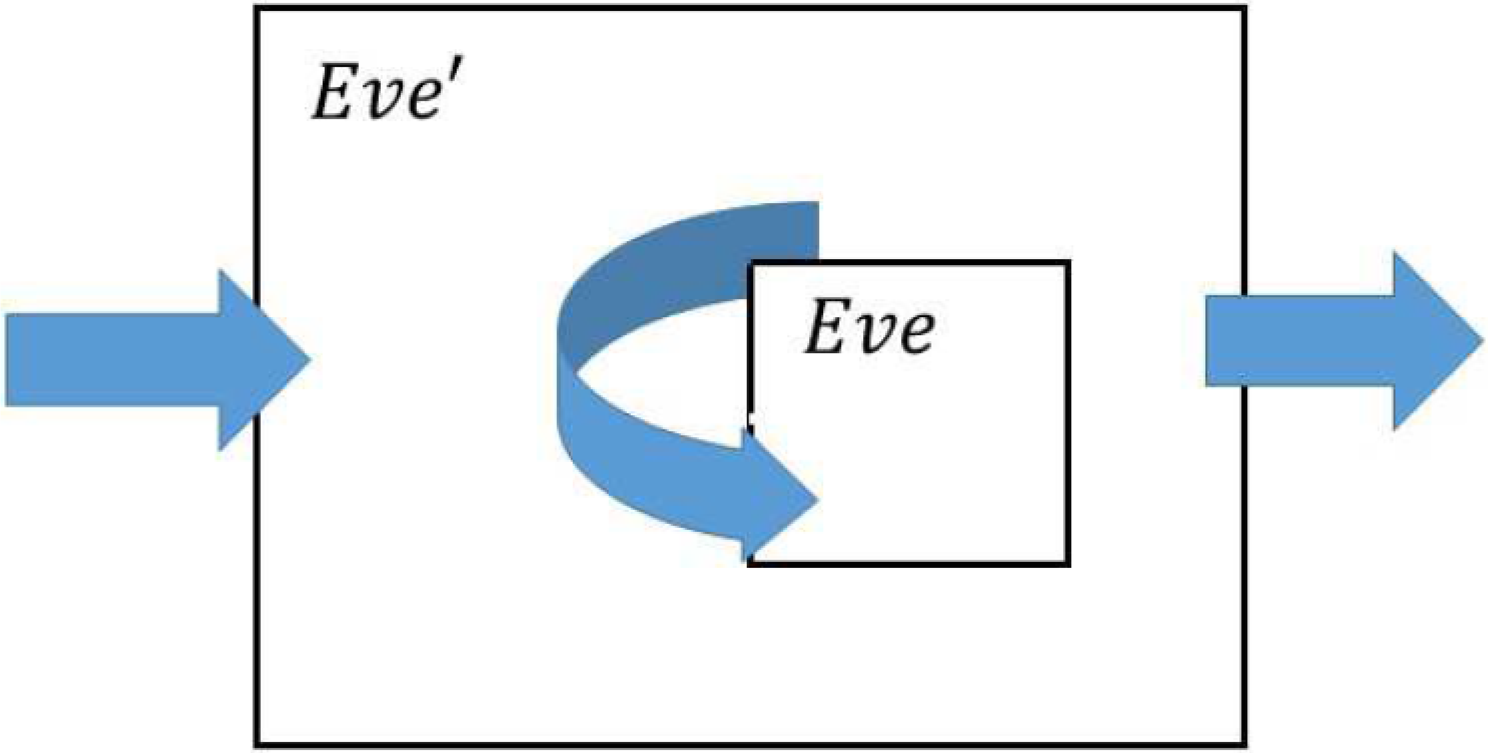
- 

Figure 2.1: We show that the security of S' implies the security of S by transforming an adversary *Eve* breaking S into an adversary *Eve*′ breaking S'

Eve breaks S → Eve' breaks S'
S' is secure → S is secure

Next Lecture

- PRG, stream ciphers ...