Perfect Security, Computational Security, and Pseudo-Random Generators

Perfect Security

Roadmap

- We will give a definition of perfect security.
  - Using a **Game!**

Roadmap

- We will give a definition of perfect security.
  - Using a **Game!** **Warning:** Most math games are not fun :-(

Roadmap

- We will give a definition of perfect security.
  - Using a **Game!** **Warning:** Most math games are not fun :-(
- Will modify for Computational Security.

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

Formal Def of Perfect Security

$\Pi = (\text{GEN}, \text{ENC}, \text{DEC})$ is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.

Formal Def of Perfect Security

$\Pi = (\text{GEN}, \text{ENC}, \text{DEC})$ is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow \mathit{GEN}$

Formal Def of Perfect Security

$\Pi = (\text{GEN}, \text{ENC}, \text{DEC})$ is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow \mathit{GEN}$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow GEN$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.
4. Alice encodes m with k: $c \leftarrow ENC_k(m)$

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow GEN$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.
4. Alice encodes m with k: $c \leftarrow ENC_k(m)$
5. Alice sends c to Eve.

Formal Def of Perfect Security

$\Pi = (\text{GEN}, \text{ENC}, \text{DEC})$ is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow GEN$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.
4. Alice encodes m with k: $c \leftarrow ENC_k(m)$
5. Alice sends c to Eve.
6. Eve outputs m_0 or m_1, hoping that her output is $DEC_k(c)$.

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow GEN$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.
4. Alice encodes m with k: $c \leftarrow ENC_k(m)$
5. Alice sends c to Eve.
6. Eve outputs m_0 or m_1, hoping that her output is $DEC_k(c)$.
7. Eve **wins** if she is right.

Formal Def of Perfect Security

Π = (GEN, ENC, DEC) is an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$.
2. Alice computes key $k \leftarrow GEN$
3. Alice chooses $m \in \{m_0, m_1\}$ randomly.
4. Alice encodes m with k: $c \leftarrow ENC_k(m)$
5. Alice sends c to Eve.
6. Eve outputs m_0 or m_1, hoping that her output is $DEC_k(c)$.
7. Eve **wins** if she is right.

Can Eve win with Prob $> \frac{1}{2}$?

Formal Def of Perfect Security

1. Easy to succeed with probability $\frac{1}{2}$

Formal Def of Perfect Security

1. Easy to succeed with probability $\frac{1}{2}$

2. Π has **perfect security** if for all Eve (algorithms)

Formal Def of Perfect Security

1. Easy to succeed with probability $\frac{1}{2}$
2. Π has **perfect security** if for all Eve (algorithms)

$$\Pr[\text{ Eve Wins}] = \frac{1}{2}$$

3. **Note:** No time or space limits on Eve.

We Want to show 1-Time Pad has Perfect Security

We Want to show 1-Time Pad has Perfect Security

We Want to show 1-Time Pad has Perfect Security

We Want to show 1-Time Pad has Perfect Security

That needs more Probability!

Detour into Probability

Conditional Probability

Conditional probability: probability that one event occurs, *given that some other event occurred*

Conditional Probability

Conditional probability: probability that one event occurs, *given that some other event occurred*

Notation: $\Pr[A|B]$.

Conditional Probability

Conditional probability: probability that one event occurs, *given that some other event occurred*

Notation: $\Pr[A|B]$.

Formal Definition: **Notation:** $\Pr[A|B] = \frac{\Pr(A \cap B)}{\Pr(B)}$.

Conditional Probability

Conditional probability: probability that one event occurs, *given that some other event occurred*

Notation: $\Pr[A|B]$.

Formal Definition: **Notation:** $\Pr[A|B] = \frac{\Pr(A \cap B)}{\Pr(B)}$.

Bayes's theorem

$$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$$

Conditional Probability

Conditional probability: probability that one event occurs, *given that some other event occurred*

Notation: $\Pr[A|B]$.

Formal Definition: **Notation:** $\Pr[A|B] = \frac{\Pr(A \cap B)}{\Pr(B)}$.

Bayes's theorem

$$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$$

Note: This is very useful in both this course and in life.

Example of Application of Bayes's theorem

$$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}.$$

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased?

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

1. Between 0.96 and 1.0

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

1. Between 0.96 and 1.0
2. Between 0.93 and 0.96

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

1. Between 0.96 and 1.0
2. Between 0.93 and 0.96
3. Between 0.90 and 0.93

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

1. Between 0.96 and 1.0
2. Between 0.93 and 0.96
3. Between 0.90 and 0.93
4. Less than 0.90.

Example of Application of Bayes's theorem

$\Pr[A|B] = \Pr[B|A] \cdot \frac{\Pr[A]}{\Pr[B]}$. There are two coins:

1) Coin F is fair: $\Pr(H) = \Pr(T) = \frac{1}{2}$.
2) Coin B is biased: $\Pr(H) = \frac{3}{4}$, $\Pr(T) = \frac{1}{4}$.

Alice picks coin at random, flips 10 times, gets all H.
Is the coin definitely biased? No.

What is Prob that it is biased? VOTE:

1. Between 0.96 and 1.0
2. Between 0.93 and 0.96
3. Between 0.90 and 0.93
4. Less than 0.90.

We will see that it is 0.982954, so between 0.98 and 0.99.

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$

$\Pr(H^{10}|B) = (\frac{3}{4})^{10}$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$

$\Pr(H^{10}|B) = (\frac{3}{4})^{10}$

$\Pr(H^{10}) = \Pr(H^{10} \cap F) + \Pr(H^{10} \cap B)$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$

$\Pr(H^{10}|B) = (\frac{3}{4})^{10}$

$\Pr(H^{10}) = \Pr(H^{10} \cap F) + \Pr(H^{10} \cap B)$

$\Pr(H^{10} \cap F) = \Pr(H^{10}|F)\Pr(F) + \Pr(H^{10}|B)\Pr(B) =$

$\frac{1}{2}\left((\frac{1}{2})^{10} + (\frac{3}{4})^{10}\right)$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$
$\Pr(H^{10}|B) = (\frac{3}{4})^{10}$
$\Pr(H^{10}) = \Pr(H^{10} \cap F) + \Pr(H^{10} \cap B)$
$\Pr(H^{10} \cap F) = \Pr(H^{10}|F)\Pr(F) + \Pr(H^{10}|B)\Pr(B) =$
$\frac{1}{2}\left((\frac{1}{2})^{10} + (\frac{3}{4})^{10}\right)$

Put it together to get

$$\Pr(B|H^{10}) = \frac{1}{1 + (2/3)^{10}} = 0.982954.$$

Example of Application of Bayes's theorem

$$\Pr(B|H^{10}) = \frac{\Pr(B)\Pr(H^{10}|B)}{P(H^{10})}$$

$\Pr(B) = \frac{1}{2}$
$\Pr(H^{10}|B) = (\frac{3}{4})^{10}$
$\Pr(H^{10}) = \Pr(H^{10} \cap F) + \Pr(H^{10} \cap B)$
$\Pr(H^{10} \cap F) = \Pr(H^{10}|F)\Pr(F) + \Pr(H^{10}|B)\Pr(B) =$
$\frac{1}{2}\left((\frac{1}{2})^{10} + (\frac{3}{4})^{10}\right)$

Put it together to get

$$\Pr(B|H^{10}) = \frac{1}{1 + (2/3)^{10}} = 0.982954.$$

$$\Pr(B|H^{n}) = \frac{1}{1 + (2/3)^{n}}.$$

Back to 1-Time Pads

1-Time Pad is Perfectly Secure

We show that Eve's Prob of winning is $\leq \frac{1}{2}$.

1-Time Pad is Perfectly Secure

We show that Eve's Prob of winning is $\leq \frac{1}{2}$.

Eve picks m_0 and m_1

1-Time Pad is Perfectly Secure

We show that Eve's Prob of winning is $\leq \frac{1}{2}$.

Eve picks m_0 and m_1

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

1-Time Pad is Perfectly Secure

We show that Eve's Prob of winning is $\leq \frac{1}{2}$.

Eve picks m_0 and m_1

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

1-Time Pad is Perfectly Secure

We show that Eve's Prob of winning is $\leq \frac{1}{2}$.

Eve picks m_0 and m_1

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

We want to show that when Eve sees $c = k \oplus m$ she can do no better than guessing to determine $m = m_0$ or $m = m_1$.

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

Eve sees $c \in \{0,1\}^n$. Knows $c = m_0 \oplus k$ OR $c = m \oplus k$ but

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

Eve sees $c \in \{0,1\}^n$. Knows $c = m_0 \oplus k$ OR $c = m \oplus k$ but

- does not know which one,

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

Eve sees $c \in \{0,1\}^n$. Knows $c = m_0 \oplus k$ OR $c = m \oplus k$ but

- does not know which one,
- does not know k.

1-Time Pad has Perfect Security

We show that Probability Eve wins is $\frac{1}{2}$

Eve picks $m_0, m_1 \in \{0,1\}^n$.

Alice flips a fair coin and picks $m \in \{m_0, m_1\}$.
Note: $\Pr(m = m_0) = \Pr(m = m_1) = \frac{1}{2}$.

Alice obtains key $k \in \{0,1\}^n$ uniformly.
Note: For any $s \in \{0,1\}^n$ $\Pr(k = s) = \frac{1}{2^n}$.

Eve sees $c \in \{0,1\}^n$. Knows $c = m_0 \oplus k$ OR $c = m \oplus k$ but

- does not know which one,
- does not know k.

Next slide completes the math.

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

$$\Pr(m = m_0 | c = m \oplus k) = \Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)}.$$

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

$$\Pr(m = m_0 | c = m \oplus k) = \Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)}.$$

$$\Pr(c = m \oplus k | m = m_0) = \Pr(c = m_0 \oplus k) = \Pr(k = c \oplus m_0) = \frac{1}{2^n}$$

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

$$\Pr(m = m_0 | c = m \oplus k) = \Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)}.$$

$$\Pr(c = m \oplus k | m = m_0) = \Pr(c = m_0 \oplus k) = \Pr(k = c \oplus m_0) = \frac{1}{2^n}$$

$$\Pr(m = m_0) = \frac{1}{2}$$

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

$$\Pr(m = m_0 | c = m \oplus k) = \Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)}.$$

$$\Pr(c = m \oplus k | m = m_0) = \Pr(c = m_0 \oplus k) = \Pr(k = c \oplus m_0) = \frac{1}{2^n}$$

$$\Pr(m = m_0) = \frac{1}{2}$$

$$\Pr(c = m \oplus k) = \Pr(k = m_0 \oplus c) = \frac{1}{2^n}$$

What Can Eve Conclude?

We want: $\Pr(m = m_0 | c = m \oplus k)$. Use Bayes Theorem

$$\Pr(m = m_0 | c = m \oplus k) = \Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)}.$$

$$\Pr(c = m \oplus k | m = m_0) = \Pr(c = m_0 \oplus k) = \Pr(k = c \oplus m_0) = \frac{1}{2^n}$$

$$\Pr(m = m_0) = \frac{1}{2}$$

$$\Pr(c = m \oplus k) = \Pr(k = m_0 \oplus c) = \frac{1}{2^n}$$

Put it all together to get

$$\Pr(c = m \oplus k | m = m_0) \frac{\Pr(m = m_0)}{\Pr(c = m \oplus k)} = \frac{1}{2^n} \frac{1/2}{1/2^n} = \frac{1}{2}$$

1-Time Pad is Perfectly Secure

1-Time Pad is Perfectly Secure

$$\Pr(m = m_0 | c = m \oplus k) = \frac{1}{2}$$

1-Time Pad is Perfectly Secure

$$\Pr(m = m_0 | c = m \oplus k) = \frac{1}{2}$$

Similarly

1-Time Pad is Perfectly Secure

$$\Pr(m = m_0 | c = m \oplus k) = \frac{1}{2}$$

Similarly

$$\Pr(m = m_1 | c = m \oplus k) = \frac{1}{2}$$

1-Time Pad is Perfectly Secure

$$\Pr(m = m_0 | c = m \oplus k) = \frac{1}{2}$$

Similarly

$$\Pr(m = m_1 | c = m \oplus k) = \frac{1}{2}$$

Hence Eve cannot get any more information. Her prob of winning is just $\frac{1}{2}$.

Shift Does Not have Perfect Security

Eve's strategy:

1. Eve chooses $m_0 = aa$ and $m_1 = ab$.
2. Alice picks one of them and encodes it with shift s. s is not known to Eve.
3. Eve gets a two-letter word.

Shift Does Not have Perfect Security

Eve's strategy:

1. Eve chooses $m_0 = aa$ and $m_1 = ab$.
2. Alice picks one of them and encodes it with shift s. s is not known to Eve.
3. Eve gets a two-letter word.
   If the letter are the same she says m_0 and is correct.

Shift Does Not have Perfect Security

Eve's strategy:

1. Eve chooses $m_0 = aa$ and $m_1 = ab$.
2. Alice picks one of them and encodes it with shift s. s is not known to Eve.
3. Eve gets a two-letter word.
   If the letter are the same she says m_0 and is correct.
   If the letter are diff she says m_1 and is correct.

Thoughts on Shift Being Insecure

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.
2. This is to be expected. Shift has a small keyspace.

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.
2. This is to be expected. Shift has a small keyspace.
3. BUT-We **did not use** that Shift had a small keyspace. What **did we use** about Shift?

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.
2. This is to be expected. Shift has a small keyspace.
3. BUT-We **did not use** that Shift had a small keyspace. What **did we use** about Shift?
   We used that it has the NY-NY problem.

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.
2. This is to be expected. Shift has a small keyspace.
3. BUT-We **did not use** that Shift had a small keyspace. What **did we use** about Shift?
   We used that it has the NY-NY problem.
4. We showed that any cipher that has NY-NY problem is computationally insecure.

Thoughts on Shift Being Insecure

1. For **Perfect Security** we allow Eve unlimited time. Later we will define **Computational Security** which restricts Eve's time. Eve used so little time for Shift, that Shift is not even computationally-secure.
2. This is to be expected. Shift has a small keyspace.
3. BUT-We **did not use** that Shift had a small keyspace. What **did we use** about Shift?
   We used that it has the NY-NY problem.
4. We showed that any cipher that has NY-NY problem is computationally insecure.

What about **Randomized Shift** . It does not have NY-NY problem. Does it have perfect Security?

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.
   If $\sigma_i = \sigma_j$ then Eve guesses m_0 and is prob correct.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.
   If $\sigma_i = \sigma_j$ then Eve guesses m_0 and is prob correct.
   If $\sigma_i \neq \sigma_j$ then Eve guesses m_1 and is definitely correct.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.
   If $\sigma_i = \sigma_j$ then Eve guesses m_0 and is prob correct.
   If $\sigma_i \neq \sigma_j$ then Eve guesses m_1 and is definitely correct.

Eve's algorithm is fast but might be wrong.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.
   If $\sigma_i = \sigma_j$ then Eve guesses m_0 and is prob correct.
   If $\sigma_i \neq \sigma_j$ then Eve guesses m_1 and is definitely correct.

Eve's algorithm is fast but might be wrong.
Probability that Eve is wrong is very low.

Randomized Shift Does Not Have Perfect Security

Does Rand Shift have Perfect Sec? **Discuss** , **Vote.** Y, N, Un.NO

Eve's strategy:

1. Eve picks $m_0 = a^{26M}$ and $m_1 = (abc \cdots z)^M$. M is large.
2. Alice returns a string

$$c = ((r_1; \sigma_1), \ldots, (r_{26M}; \sigma_{26M})).$$

3. (Informal argument) M is chosen large enough so that with high probability two of the r's the same and NOT 26 apart, say r_i and r_j.
   If $\sigma_i = \sigma_j$ then Eve guesses m_0 and is prob correct.
   If $\sigma_i \neq \sigma_j$ then Eve guesses m_1 and is definitely correct.

Eve's algorithm is fast but might be wrong.
Probability that Eve is wrong is very low.
This motivates the definition of Comp Security which we give later.

An Instructive Example

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.
2. Alice use the recurrence

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.
2. Alice use the recurrence
   $a_0 = b_0$

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.
2. Alice use the recurrence
   $a_0 = b_0$
   $a_1 = b_1$

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.
2. Alice use the recurrence
   $a_0 = b_0$
   $a_1 = b_1$
   $(\forall n \geq 2)[a_n = Aa_{n-1} + Ba_{n-2}]$

An Interesting Complicated Insecure Cipher

Alice encodes as follows:

1. Alice pick random key $(b_0, b_1, A, B) \in \{0,1\}^4$.
2. Alice use the recurrence
   $a_0 = b_0$
   $a_1 = b_1$
   $(\forall n \geq 2)[a_n = Aa_{n-1} + Ba_{n-2}]$
   to generate a sequence and use this as a 1-time pad.

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \qquad a_1 = 1,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \qquad a_1 = 1, \qquad a_2 = 1,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \qquad a_1 = 1, \qquad a_2 = 1,$

$a_3 = 0,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0, \quad a_7 = 1,$

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0, \quad a_7 = 1, \quad a_8 = 1.$

So the pattern is $(011)^*$.

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0, \quad a_7 = 1, \quad a_8 = 1.$

So the pattern is $(011)^*$.

To encode 001100 would XOR with 011011 to get 010111.

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0, \quad a_7 = 1, \quad a_8 = 1.$

So the pattern is $(011)^*$.

To encode 001100 would XOR with 011011 to get 010111.

All keys (b_0, b_1, A, B) lead to a pattern.

An Instructive Example

Example $b_0 = 0,\ b_1 = 1,\ A = 1,\ B = 1.$

$a_0 = 0, \quad a_1 = 1, \quad a_2 = 1,$

$a_3 = 0, \quad a_4 = 1, \quad a_5 = 1.$

$a_6 = 0, \quad a_7 = 1, \quad a_8 = 1.$

So the pattern is $(011)^*$.

To encode 001100 would XOR with 011011 to get 010111.

All keys (b_0, b_1, A, B) lead to a pattern.

We present the patterns on the next slide.

Which Keys Give Which Patterns

(b_0, b_1, A, B)	pattern
$(0, 0, A, B)$	0^*
$(0, 1, 0, 0)$	010^*
$(1, 0, 0, 0)$	100^*
$(1, 1, 0, 0)$	110^*
$(0, 1, 0, 1)$	$(01)^*$
$(1, 0, 0, 1)$	$(10)^*$
$(1, 1, 0, 1)$	1^*
$(0, 1, 1, 0)$	01^*
$(1, 0, 1, 0)$	10^*
$(1, 1, 1, 0)$	$(101)^*$
$(0, 1, 1, 1)$	$(011)^*$
$(1, 0, 1, 1)$	$(101)^*$
$(1, 1, 1, 1)$	$(110)^*$

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Eve should use m_0, m_1 of length 6.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Eve should use m_0, m_1 of length 6.

She'll try $m_0 = 000111$ and $m_1 = 111000$.

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Eve should use m_0, m_1 of length 6.

She'll try $m_0 = 000111$ and $m_1 = 111000$.

Next slide is

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Eve should use m_0, m_1 of length 6.

She'll try $m_0 = 000111$ and $m_1 = 111000$.

Next slide is

- Encodings of 000111

Helping Eve Pick an m_0 and m_1

The patterns are in three categories:

- Eventually periodic of length 1: 0^*, 010^*, 100^*, 1^*, 01^*, 10^*.
- Periodic of length 2: $(01)^*$, $(10)^*$.
- Periodic of length 3: $(101)^*$, $(011)^*$, $(101)^*$, $(110)^*$.

The least common multiple of $1, 2, 3$ is 6.

Eve should use m_0, m_1 of length 6.

She'll try $m_0 = 000111$ and $m_1 = 111000$.

Next slide is

- Encodings of 000111
- Encodings of 111000

Encoding 000111 and 111000

(b_0, b_1, A, B)	pattern	$E(000111)$	$E(111000)$
$(0, 0, A, B)$	0^*	000111	111000
$(0, 1, 0, 0)$	010^*	010111	101000
$(1, 0, 0, 0)$	100^*	100111	011000
$(1, 1, 0, 0)$	110^*	110111	001000
$(0, 1, 0, 1)$	$(01)^*$	010010	101101
$(1, 0, 0, 1)$	$(10)^*$	101101	010010
$(1, 1, 0, 1)$	1^*	111000	000111
$(0, 1, 1, 0)$	01^*	011000	100111
$(1, 0, 1, 0)$	10^*	100111	011000
$(1, 1, 1, 0)$	$(101)^*$	101010	010101
$(0, 1, 1, 1)$	$(011)^*$	011100	100011
$(1, 0, 1, 1)$	$(101)^*$	101010	010101
$(1, 1, 1, 1)$	$(110)^*$	110001	001110

If Eve Sees... Then Guess 000111

Eve Sees	Numb Keys yield 000111	Numb Keys Yield 111000
000111	4	1
010111	1	0
100111	2	0
110111	1	0
101101	1	1
101010	1	0
011100	1	0
101010	1	0
011100	1	0

If Eve Sees... Then Guess 000111

Eve Sees	Numb Keys yield 000111	Numb Keys Yield 111000
000111	4	1
010111	1	0
100111	2	0
110111	1	0
101101	1	1
101010	1	0
011100	1	0
101010	1	0
011100	1	0

Number of keys that lead to the Eve being right:

$$4 + 1 + 2 + 1 + 1 + 1 + 1 + 1 + 1 = 13.$$

If Eve Sees... Then Guess 111000

Eve Sees	Numb Keys yield 000111	Numb Keys Yield 111000
010010	1	2
111000	1	4
011000	1	2
101000	0	1
001000	0	1
100111	0	1
100011	0	1
010101	0	1
001110	0	1

If Eve Sees... Then Guess 111000

Eve Sees	Numb Keys yield 000111	Numb Keys Yield 111000
010010	1	2
111000	1	4
011000	1	2
101000	0	1
001000	0	1
100111	0	1
100011	0	1
010101	0	1
001110	0	1

Number of keys that lead to the Eve being right:

$$2 + 4 + 2 + 1 + 1 + 1 + 1 + 1 + 1 = 14.$$

Prob that Eve Wins

The prob that Eve wins is the sum of the following:

Prob that Eve Wins

The prob that Eve wins is the sum of the following:

1. $\frac{1}{2} \times$ Pr Alice picks 000111 and one of the 13 keys where Eve wins

$$= \frac{1}{2} \times \frac{13}{16} = \frac{13}{32}.$$

Prob that Eve Wins

The prob that Eve wins is the sum of the following:

1. $\frac{1}{2} \times$ Pr Alice picks 000111 and one of the 13 keys where Eve wins

$$= \frac{1}{2} \times \frac{13}{16} = \frac{13}{32}.$$

2. $\frac{1}{2} \times$ Pr Alice picks 111000 and one of the 14 keys where Eve wins

$$= \frac{1}{2} \times \frac{14}{16} = \frac{14}{32}.$$

Prob that Eve Wins

The prob that Eve wins is the sum of the following:

1. $\frac{1}{2} \times$ Pr Alice picks 000111 and one of the 13 keys where Eve wins

$$= \frac{1}{2} \times \frac{13}{16} = \frac{13}{32}.$$

2. $\frac{1}{2} \times$ Pr Alice picks 111000 and one of the 14 keys where Eve wins

$$= \frac{1}{2} \times \frac{14}{16} = \frac{14}{32}.$$

The prob that Eve wins is

$$\frac{13}{32} + \frac{14}{32} = \frac{27}{32} \sim 0.84.$$

Prob that Eve Wins

The prob that Eve wins is the sum of the following:

1. $\frac{1}{2} \times$ Pr Alice picks 000111 and one of the 13 keys where Eve wins

$$= \frac{1}{2} \times \frac{13}{16} = \frac{13}{32}.$$

2. $\frac{1}{2} \times$ Pr Alice picks 111000 and one of the 14 keys where Eve wins

$$= \frac{1}{2} \times \frac{14}{16} = \frac{14}{32}.$$

The prob that Eve wins is

$$\frac{13}{32} + \frac{14}{32} = \frac{27}{32} \sim 0.84.$$

Hence the cipher is insecure since this is $> \frac{1}{2}$.

Computational Security

Computational Security

- Idea: Relax perfect indistinguishability
  1. Allow Eve to win with probability slightly more than $\frac{1}{2}$.
  2. Bound how powerful Eve is.
- Two approaches
  - Concrete security (we omit)
  - Asymptotic security

Asymptotic security

- Introduce **security parameter** n
  - For now, can view as the key length
  - Fixed by honest parties at initialization
    - Allows users to tailor the security level
  - Known by Eve
- Measure running times of all parties, and the success probability of Eve, as functions of n

Computational Security (asymptotic)

- Computational Security
  - Security may fail with probability **negligible in n**
  - Eve's algorithm is allowed to be randomized but must stop in poly time. PPT=Prob Poly Time.

Note: A Randomized Algorithm is allowed to flip coins but has a small prob of being wrong. Considered a good definition of efficiency.

Definitions

- A function $f : \mathbb{Z}^+ \to \mathbb{Z}^+$ is (at most) **polynomial** if there exists c such that $f(n) < n^c$ for large enough n
- A function $f : \mathbb{Z}^+ \to [0,1]$ is **negligible** if for every polynomial p is holds that $f(n) < \frac{1}{p(n)}$ for large enough n
  - Typical example: $f(n) = poly(n) \cdot 2^{-cn}$

Notation: Denote polynomial by **poly** . Denote negligible by **neg**

.

Why these choices?

- Taking **Efficient** to mean PPT is standard.
- poly and neg both have convenient closure properties
  - poly * poly = poly
    - Poly-many calls to PPT subroutine (with poly-size inputs) is PPT
  - poly * neg = neg
    - Poly-many calls to subroutine that fails with neg probability fails with neg probability overall

Eve is Less Powerful. What About Alice and Bob?

When we first defined an encryption scheme (GEN, ENC_k, DEC_k) we did not mention that in order for Alice and Bob to use the scheme:

1. GEN had to be fast to compute.
2. ENC_k had to be fast to compute.
3. DEC_k had to be fast to compute.

This was informally true of Shift, ..., RSA. But could not formalize since we did not have a security parameter n.

We will now redefine Encryption Scheme with a security parameter n and formalize that GEN, ENC_k, DEC_k must be fast to compute.

(Re)defining encryption

- A **private-key encryption scheme** is defined by three **PPT** algorithms (GEN, ENC, DEC):
  - **GEN:** takes as input $\mathbf{1^n}$; outputs k. (assumed $|k| \geq n$).
  - **ENC:** takes as input a key k and a message $m \in \{0,1\}^x$; outputs ciphertext c

$$c \leftarrow ENC_k(m)$$

  - **DEC:** takes key k and ciphertext c as input; outputs a message m or "error"

Computational Security

$\Pi = (\text{GEN}, \text{ENC}, \text{DEC})$ an enc sch. Message space $\mathcal{M}$.

Game: Alice and Eve are the players. Alice has full access to Π.

1. Eve chooses $m_0, m_1 \in \mathcal{M}$, $|m_0| = |m_1|$
2. Alice computes $k \leftarrow GEN(1^n)$
3. Alice picks $m \in \{m_0, m_1\}$
4. Alice encodes m: $c \leftarrow ENC_k(m)$
5. Alice sends c to Eve.
6. Eve outputs m_0 or m_1, hoping that her output is $DEC_k(c)$.
7. Eve **wins** if she is right.

We discuss Eve's runtime and Prob of winning on the next slide.

Computational Security

Π is **computationally secure** if for all **PPT** Eve, there is a **neg function** $\epsilon(n)$ such that

$$\Pr[\text{Eve Wins}] \leq \frac{1}{2} + \epsilon(\mathbf{n})$$

Intuition Behind Example We Will Do

Fact: The only Encryption scheme with **perfect security** is 1-time pad. But that is hard to use (randomness expensive).

What to Do?: Computational Sec. instead of Perfect Sec.

Pseudorandomness: Rather than demand **perfect** randomness for 1-time pad we settle for **pseudorandomness** .

1. Psuedorandom: Intuitively means that to a poly-bounded Eve the sequence looks random.
2. Using psuedorandom instead of random:
   - 2.1 **PRO:** Practical! Being Used!
   - 2.2 **CON:** Won't get perfect secrecy.

Pseudorandomness

What does “uniform” mean?

Which of the following is a uniform string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

What does "uniform" mean?

Which of the following is a uniform string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

Trick Question! There is no such think as a uniform **string** .
There is the uniform **Distribution** .
Def: The **uniform dist** on $\{0,1\}^n$ picks each string with prob $\frac{1}{2^n}$.

Pseudorandom generators (PRGs)

- A PRG is an efficient, deterministic algorithm that expands a **short, uniform seed** into a **longer, pseudorandom output**
  - Useful whenever you have a "small" number of true random bits, and want lots of "random-looking" bits

Definition of PRGs

Let $G : \{0,1\}^n \to \{0,1\}^{p(n)}$ be an efficient, deterministic algorithm.

Game: Alice and Eve are the players. Both have access to G.

1. Alice picks $x \in \{0,1\}^n$ unif, computes $y = G(x) \in \{0,1\}^{p(n)}$.
2. Alice picks $z \in \{0,1\}^{p(n)}$ unif.
3. Alice gives y, z to Eve
4. Eve tries to determine which one is $G(x)$ and which one is not.
5. Eve says which is which. If she is right she wins!

Can Eve win this game with probability over $\frac{1}{2}$?

Definition of PRGs

G is a PRG if for all **PPT** Eves, there is a **neg function** $\epsilon(n)$ such that

$$\Pr[\text{Eve Wins}] \leq \frac{1}{2} + \epsilon(\mathbf{n})$$

Candidate for a PRG

D_n will be the strings in $\{0,1\}^{n^2}$ that come out of the following process.

1. Pick safe prime p, length n ($\{0,\ldots,p-1\}$ has $\sim 2^n$ elts).
2. Find a generator g for p of length n.
3. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
4. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
5. Let b_i be the right-most-bit g^i.
6. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
Assuming Discrete Log is hard

Candidate for a PRG

D_n will be the strings in $\{0,1\}^{n^2}$ that come out of the following process.

1. Pick safe prime p, length n ($\{0,\ldots,p-1\}$ has $\sim 2^n$ elts).
2. Find a generator g for p of length n.
3. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
4. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
5. Let b_i be the right-most-bit g^i.
6. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
Assuming Discrete Log is hard still not known.

Candidate for a PRG

D_n will be the strings in $\{0,1\}^{n^2}$ that come out of the following process.

1. Pick safe prime p, length n ($\{0,\ldots,p-1\}$ has $\sim 2^n$ elts).
2. Find a generator g for p of length n.
3. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
4. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
5. Let b_i be the right-most-bit g^i.
6. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
Assuming Discrete Log is hard still not known.
But thought to be PRG.

Candidate for a PRG

D_n will be the strings in $\{0,1\}^{n^2}$ that come out of the following process.

1. Pick safe prime p, length n ($\{0,\ldots,p-1\}$ has $\sim 2^n$ elts).
2. Find a generator g for p of length n.
3. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
4. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
5. Let b_i be the right-most-bit g^i.
6. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
Assuming Discrete Log is hard still not known.
But thought to be PRG. At least by me.

Do PRGs exist?

Do PRGs exist?

1. We don't know

Do PRGs exist?

1. We don't know . . . Would imply P $\neq$ NP.

Do PRGs exist?

1. We don't know . . . Would imply P $\neq$ NP.
2. We will *assume* certain algorithms are PRGs.

Do PRGs exist?

1. We don't know . . . Would imply P $\neq$ NP.
2. We will *assume* certain algorithms are PRGs.
3. Can *construct* PRGs from weaker assumptions. (We will not do this.)

Using Pseudo one-time pad

- Let G be a deterministic algorithm, with $|G(k)| = p(|k|)$
- $Gen(1^n)$: output uniform n-bit key k
  - Security parameter $n \Rightarrow$ message space $\{0,1\}^{p(n)}$
- $Enc_k(m)$: output $G(k) \oplus m$
- $Dec_k(n)$: output $G(k) \oplus c$
- correctness is obvious

Security of pseudo-OTP?

Theorem: Pseudo-OTP is comp secure.

Proof Sketch: Can show that if not comp secure then G is not PRG. We omit details.

Stepping back

- *Proof* that the pseudo OTP is secure . . .
- . . . with some caveats
  - Assuming G is a pseudorandom generator
  - Relative to our definition
- The *Only* way the scheme can be broken is:
  - If a weakness is found in G
  - If the definition isn't sufficiently strong . . .

Have we gained anything?

- YES: the pseudo-OTP has a key shorter than the message
  - n bits vs. p(n) bits
- Practitioners know what to aim for: Good Psuedo-random generators.