Pseudorandomness

What Does RANDOM Mean?

Which of the following is a RANDOM string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

What Does RANDOM Mean?

Which of the following is a RANDOM string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

Trick Question! There is no such think as a random **string** .

What Does RANDOM Mean?

Which of the following is a RANDOM string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

Trick Question! There is no such think as a random **string** . There is the uniform **dist** where all strings are equally likely.

What Does RANDOM Mean?

Which of the following is a RANDOM string?

- 0101010101010101
- 0010111011100110
- 0000000000000000

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

Pseudorandom generators (PRGs)

Pseudorandom generators (PRGs)

1. **Informal Definition** A PRG is a poly, algorithm that expands a **short, uniform seed** into a **longer, output** that is hard to distinguish from random.

Pseudorandom generators (PRGs)

1. **Informal Definition** A PRG is a poly, algorithm that expands a **short, uniform seed** into a **longer, output** that is hard to distinguish from random.
2. Useful for psuedo One Time Pad.

Pseudorandom generators (PRGs)

1. **Informal Definition** A PRG is a poly, algorithm that expands a **short, uniform seed** into a **longer, output** that is hard to distinguish from random.
2. Useful for psuedo One Time Pad.
3. The Keyword-shift cipher was a primitive example.

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Let p be a polynomial.

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Let p be a polynomial.
Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Let p be a polynomial.
Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

Our intent is that $G(x)$ **looks random** .

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Let p be a polynomial.
Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

Our intent is that $G(x)$ **looks random** .

$p(n) \geq n^2$. If (say) $p(n) = n+1$ then that is not helpful.

Intro to Formal Definition of PRGs

We define what a PRG is formally using **A Game!**

My wife says that **Math Games** are not **Fun Games**

The definition of PRG using games is evidence of her assertion.

Let p be a polynomial.
Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

Our intent is that $G(x)$ **looks random** .

$p(n) \geq n^2$. If (say) $p(n) = n+1$ then that is not helpful.
Might be on HW. Might be a HS student project :-)

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
Game: Alice and Eve are the players. Both have access to G.

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
Game: Alice and Eve are the players. Both have access to G.

1. Alice picks $x \in \{0,1\}^n$ **unif** , computes $y = G(x) \in \{0,1\}^{p(n)}$.

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

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** .

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.

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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.
4. Eve outputs one of $\{w_1, w_2\}$ hoping its z.

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.
4. Eve outputs one of $\{w_1, w_2\}$ hoping its z.
5. If Eve output z then she wins!

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.
4. Eve outputs one of $\{w_1, w_2\}$ hoping its z.
5. If Eve output z then she wins!

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

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.
4. Eve outputs one of $\{w_1, w_2\}$ hoping its z.
5. If Eve output z then she wins!

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

Formal Definition of Game Associated with PRGs

Let $G : \{0,1\}^n \rightarrow \{0,1\}^{p(n)}$ be computable in poly time.
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 $\{w_1, w_2\} = \{y, z\}$ to Eve. z is either w_1 or w_2.
4. Eve outputs one of $\{w_1, w_2\}$ hoping its z.
5. If Eve output z then she wins!

Can Eve win this game with probability over $\frac{1}{2}$? Discuss.
Depends on how much Computational Power Eve has.

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

(This takes Exponential Time!)

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)
3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)
3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**
4. If $w_1 \notin A$ then Eve outputs w_1 and **she is right!**

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)
3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**
4. If $w_1 \notin A$ then Eve outputs w_1 and **she is right!**
5. If $w_1, w_2 \in A$ then Eve outputs w_1. She might be wrong.

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)
3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**
4. If $w_1 \notin A$ then Eve outputs w_1 and **she is right!**
5. If $w_1, w_2 \in A$ then Eve outputs w_1. She might be wrong.

Prob that Eve loses is $\leq$ Prob that $z \in A$.

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)

3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**
4. If $w_1 \notin A$ then Eve outputs w_1 and **she is right!**
5. If $w_1, w_2 \in A$ then Eve outputs w_1. She might be wrong.

Prob that Eve loses is $\le$ Prob that $z \in A$.
There are $2^{p(n)}$ strings that z could be. Only 2^n of them are in A.

Eve Wins with prob $> \frac{1}{2}$ with Unlimited Comp

Eve's strategy:

1. Eve gets as input w_1, w_2. z is either w_1 or w_2.
2. Eve creates the set

$$A = \{G(x) : x \in \{0,1\}^n\}$$

   (This takes Exponential Time!)
3. If $w_2 \notin A$ then Eve outputs w_2 and **she is right!**
4. If $w_1 \notin A$ then Eve outputs w_1 and **she is right!**
5. If $w_1, w_2 \in A$ then Eve outputs w_1. She might be wrong.

Prob that Eve loses is $\le$ Prob that $z \in A$.
There are $2^{p(n)}$ strings that z could be. Only 2^n of them are in A.
Prob Eve loses is $\le$ prob $z \in A$ which is $\frac{2^n}{2^{p(n)}} = \frac{1}{2^{p(n)-n}} < \frac{1}{2}$.

Psuedo Random Only Needs to Fool Poly Time Eve

Psuedo Random Only Needs to Fool Poly Time Eve

1. In our def of PRG we will give Eve only poly time.

Psuedo Random Only Needs to Fool Poly Time Eve

1. In our def of PRG we will give Eve only poly time.
2. We will allow Eve to use randomization.

Psuedo Random Only Needs to Fool Poly Time Eve

1. In our def of PRG we will give Eve only poly time.
2. We will allow Eve to use randomization.
3. We will even allow Eve to be right $> \frac{1}{2}$ of the time, but not much bigger.

Definitions Needed For PRG

Definitions Needed For PRG

1. A function $f : \mathbb{Z}^+ \to [0,1]$ is **negligible** if, for every poly p, for large n, $f(n) < \frac{1}{p(n)}$. We use **neg.** **Example** $f(n) = \frac{1}{2^n}$

Definitions Needed For PRG

1. A function $f : \mathbb{Z}^+ \rightarrow [0,1]$ is **negligible** if, for every poly p, for large n, $f(n) < \frac{1}{p(n)}$. We use **neg.** **Example** $f(n) = \frac{1}{2^n}$
2. An algorithm is **Poly Prob Time (PPT)** if there is a randomized alg for it that halts in poly time and has a **neg** prob of error. **Example** Primality.

Formal Definition of PRGs (Finally!)

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

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

Candidate for a PRG

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.
7. Output $b_1 b_2 \cdots b_{n^2}$

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.
7. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.
7. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
But assuming Discrete Log is hard

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.
7. Output $b_1 b_2 \cdots b_{n^2}$

Not known if this is really PRG.
But assuming Discrete Log is hard still not known!

Candidate for a PRG

1. Input $b \in \{0,1\}^n$.
2. Find p, be the first safe prime $\geq 1b$.
3. Find g, the smallest generator of of $\mathbb{Z}_p^*$. Note that $\mathbb{Z}_p^*$ has $\sim 2^n$ elements and every element of it can be viewed as an n-bit string.
4. Compute $(g^1, g^2, \ldots, g^{n^2})$ all mod p.
5. View $(g^1, g^2, \ldots, g^{n^2})$ as n-bit strings.
6. Let b_i be the right-most-bit of g^i.
7. Output $b_1 b_2 \cdots b_{n^2}$

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

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. People **assume** certain functions are PRGs.

Do PRGs exist?

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

Pseudo One-Time Pad

Let G be a PRG from $\{0,1\}^n$ to $\{0,1\}^{p(n)}$.

Pseudo One-Time Pad

Let G be a PRG from $\{0,1\}^n$ to $\{0,1\}^{p(n)}$.

1. Psuedo One-Time Pad:

Pseudo One-Time Pad

Let G be a PRG from $\{0,1\}^n$ to $\{0,1\}^{p(n)}$.

1. Psuedo One-Time Pad:
2. Alice generates an n-bit string k, n truely random bits.

Pseudo One-Time Pad

Let G be a PRG from $\{0,1\}^n$ to $\{0,1\}^{p(n)}$.

1. Psuedo One-Time Pad:
2. Alice generates an n-bit string k, n truely random bits.
3. Alice computer $G(k) = k'$, $p(n)$ psuedo-random bits.

Pseudo One-Time Pad

Let G be a PRG from $\{0,1\}^n$ to $\{0,1\}^{p(n)}$.

1. Psuedo One-Time Pad:
2. Alice generates an n-bit string k, n truely random bits.
3. Alice computer $G(k) = k'$, $p(n)$ psuedo-random bits.
4. Alice and Bob use k' for their 1-time pad.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.

PRO Info-Theoretic Secure.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.

PRO Info-Theoretic Secure.

CON Generating truly random bits is hard.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.

PRO Info-Theoretic Secure.

CON Generating truly random bits is hard.

Used This really has been used, but only for short messages where security is crucial. The US-Russia Red Phone.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.
PRO Info-Theoretic Secure.
CON Generating truly random bits is hard.
Used This really has been used, but only for short messages where security is crucial. The US-Russia Red Phone.

Psuedo One-Time Pad One can define Comp-theoretic security rigorously. With that definition, one can show that the Psuedo One-Time Pad is comp-theoretic secure.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.
PRO Info-Theoretic Secure.
CON Generating truly random bits is hard.
Used This really has been used, but only for short messages where security is crucial. The US-Russia Red Phone.

Psuedo One-Time Pad One can define Comp-theoretic security rigorously. With that definition, one can show that the Psuedo One-Time Pad is comp-theoretic secure.
PRO Comp-Theoretic Secure.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.
PRO Info-Theoretic Secure.
CON Generating truly random bits is hard.
Used This really has been used, but only for short messages where security is crucial. The US-Russia Red Phone.

Psuedo One-Time Pad One can define Comp-theoretic security rigorously. With that definition, one can show that the Psuedo One-Time Pad is comp-theoretic secure.
PRO Comp-Theoretic Secure.
CON Proving that G is a PRG is hard.

One Time-Pad/Psuedo One-Time Pad

One-Time Pad One can define info-theoretic security rigorously. With that definition, one can show that the One-Time Pad is info-theoretic secure.

PRO Info-Theoretic Secure.

CON Generating truly random bits is hard.

Used This really has been used, but only for short messages where security is crucial. The US-Russia Red Phone.

Psuedo One-Time Pad One can define Comp-theoretic security rigorously. With that definition, one can show that the Psuedo One-Time Pad is comp-theoretic secure.

PRO Comp-Theoretic Secure.

CON Proving that G is a PRG is hard.

Used This is used, but with functions G that seem like PRGs but there is no proof of that.

One Time-Pad/Psuedo One-Time Pad

Both One-Time Pad and Psuedo One-Time Pad have a problem for usage:

One Time-Pad/Psuedo One-Time Pad

Both One-Time Pad and Psuedo One-Time Pad have a problem for usage:

Once the block of bits runs out, you can't communicate anymore!

One Time-Pad/Psuedo One-Time Pad

Both One-Time Pad and Psuedo One-Time Pad have a problem for usage:

Once the block of bits runs out, you can't communicate anymore!

Hence, rather than use either One-Time Pads or Psuedo One-Time Pads, we use

One Time-Pad/Psuedo One-Time Pad

Both One-Time Pad and Psuedo One-Time Pad have a problem for usage:

Once the block of bits runs out, you can't communicate anymore!

Hence, rather than use either One-Time Pads or Psuedo One-Time Pads, we use

Stream Ciphers

One Time-Pad/Psuedo One-Time Pad

Both One-Time Pad and Psuedo One-Time Pad have a problem for usage:

Once the block of bits runs out, you can't communicate anymore!

Hence, rather than use either One-Time Pads or Psuedo One-Time Pads, we use

Stream Ciphers

which is the next lecture.