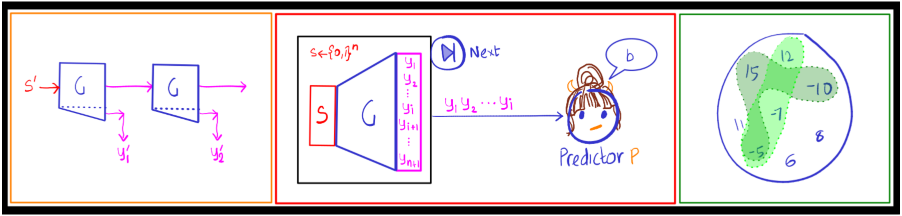


CS409m: Introduction to Cryptography

Lecture 06 (20/Aug/25)

Instructor: Chethan Kamath

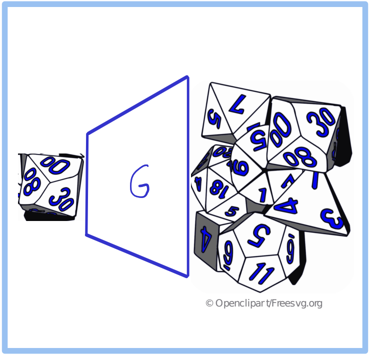
Recall from Previous Lecture...

- Task: secure communication of *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

Recall from Previous Lecture...

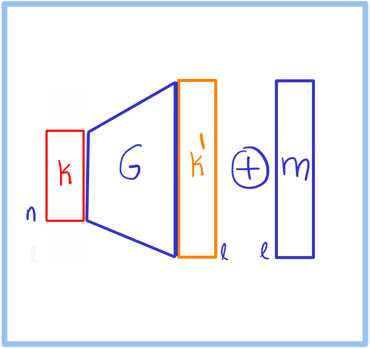
- Task: secure communication of *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

Pseudorandom Generator (PRG)



© Openclipart/Freesvg.org

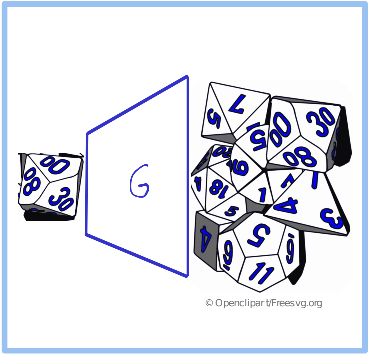
Computational One-Time Pad



Recall from Previous Lecture...

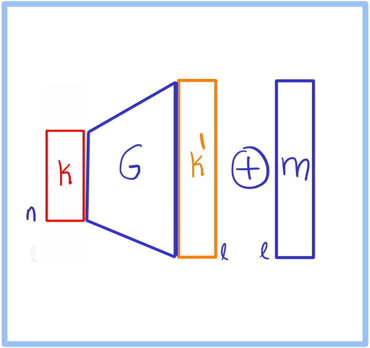
- Task: secure communication of *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

Pseudorandom Generator (PRG)



© Openclipart/Freesvg.org

Computational One-Time Pad



Main tool: proof by reduction

Recall from Previous Lecture...

Recall PRG: expanding function whose output is computationally indistinguishable from uniformly random

Recall from Previous Lecture...

Recall PRG: expanding function whose output is computationally indistinguishable from uniformly random

Definition 1 (Two-worlds definition)

Let G be an efficient deterministic algorithm that for any $n \in \mathbb{N}$ and input $s \in \{0,1\}^n$, outputs a string of length $\ell(n) > n$.
G is PRG if for every PPT *distinguisher* D

$$\delta(n) := \left| \Pr_{s \leftarrow \{0,1\}^n}[\mathsf{D}(G(s)) = 0] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[\mathsf{D}(r) = 0] \right|$$

is negligible.

Recall from Previous Lecture...

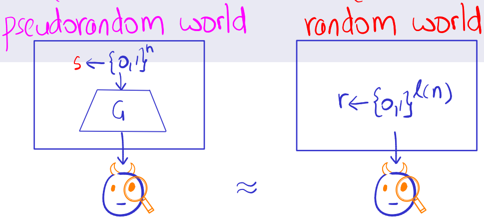
Recall PRG: expanding function whose output is computationally indistinguishable from uniformly random

Definition 1 (Two-worlds definition)

Let G be an efficient deterministic algorithm that for any $n \in \mathbb{N}$ and input $s \in \{0,1\}^n$, outputs a string of length $\ell(n) > n$.
G is PRG if for every PPT *distinguisher* D

$$\delta(n) := \left| \Pr_{s \leftarrow \{0,1\}^n}[\mathsf{D}(G(s)) = 0] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[\mathsf{D}(r) = 0] \right|$$

is negligible.



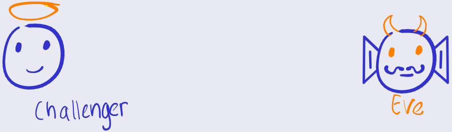
Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.

SKE World



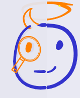
Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. $\exists$D for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.

SKE World



Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.

SKE World

Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

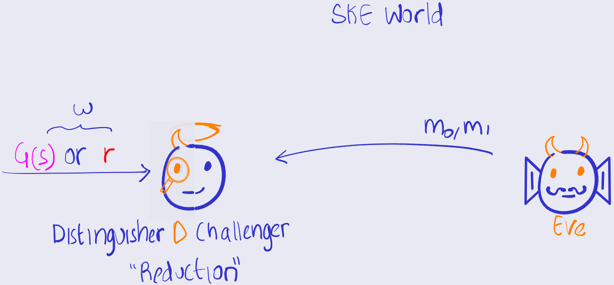
Proof by reduction. $\exists D$ for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.

Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. $\exists$D for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.

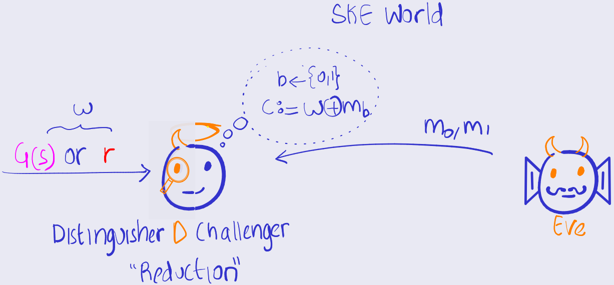


Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.

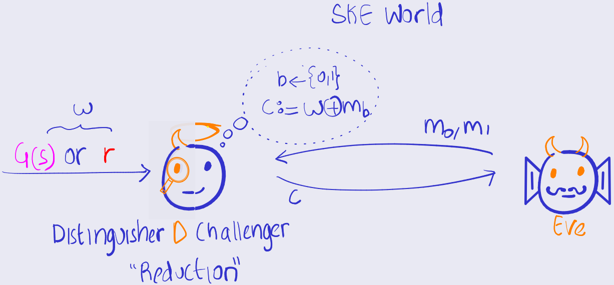


Recall from Previous Lecture...

Theorem 1

If G *is a PRG, then Comp. OTP is comp. secret against eavesdroppers*

Proof by reduction. $\exists$D for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.

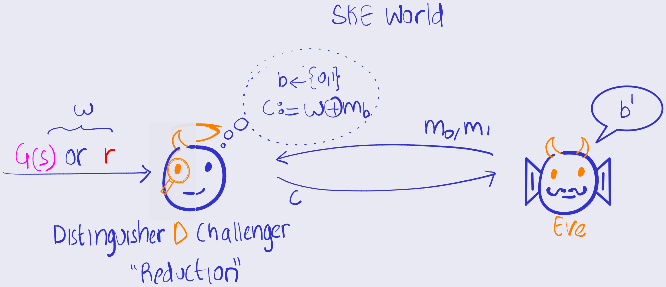


Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.

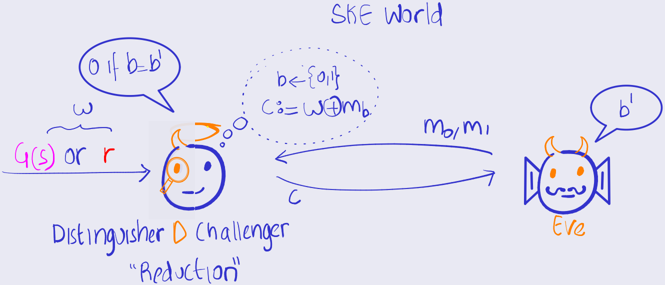


Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.

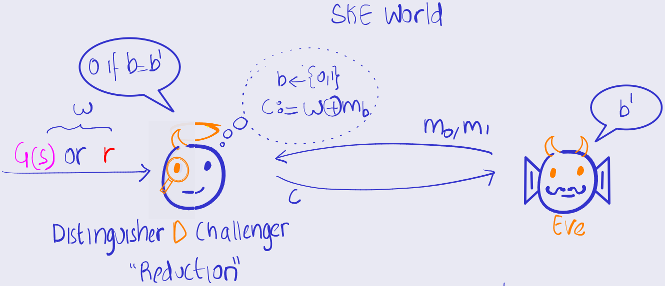


Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. $\exists$D for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.



Analysis

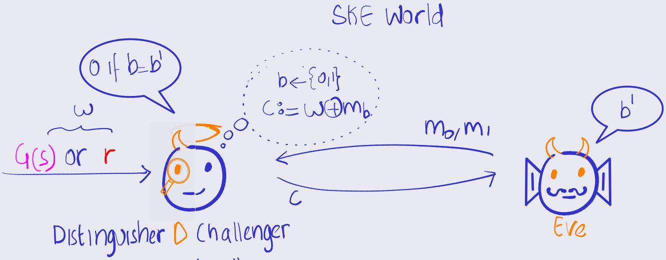
We want to show: $\left| \Pr_{s\leftarrow\{0,1\}^n}[D(G(s))=0] - \Pr_{r\leftarrow\{0,1\}^{\ell(n)}}[D(r)=0] \right|$ not negl

Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.



"Reduction"

Analysis

We want to show: $\left| \Pr_{s\leftarrow\{0,1\}^n}[D(G(s))=0] - \Pr_{r\leftarrow\{0,1\}^{\ell(n)}}[D(r)=0] \right|$ not negl

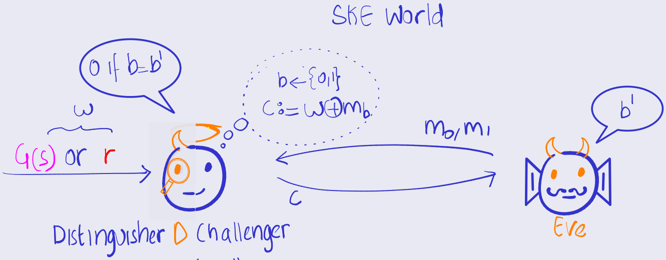
$= \Pr[Eve(c)=b]$ for Construction 2

Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. ∃D for G ⇐ ∃Eve breaking Computational OTP.



Analysis

"Reduction"

We want to show: $\left| \Pr_{s \leftarrow \{0,1\}^n}[D(G(s))=0] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[D(r)=0] \right|$ not negl

$= \Pr[Eve(c)=b]$ for Construction 2

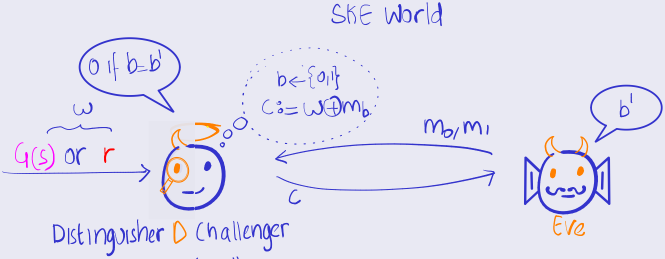
$= \frac{1}{2}$ + non-negl

Recall from Previous Lecture...

Theorem 1

If G is a PRG, then Comp. OTP is comp. secret against eavesdroppers

Proof by reduction. $\exists$D for G $\Leftarrow$ $\exists$Eve breaking Computational OTP.



Analysis

"Reduction"

We want to show: $\left| \Pr_{s\leftarrow\{0,1\}^n}[D(G(s))=0] - \Pr_{r\leftarrow\{0,1\}^{\ell(n)}}[D(r)=0] \right|$ not negl

$= \Pr[Eve(c)=b]$ for construction 2

$$\left| \cancel{\tfrac{1}{2}} + \text{non-negl} - \cancel{\tfrac{1}{2}} \right| = \Pr[Eve(c)=b] \text{ for OTP}$$

Plan for Today's Lecture...

- Task: secure communication of *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

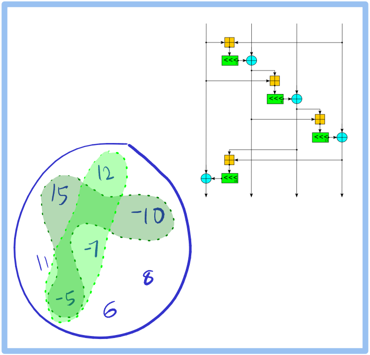
Plan for Today's Lecture...

- Task: secure communication of very *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

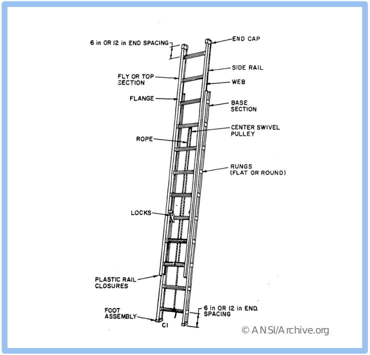
Plan for Today's Lecture...

- Task: secure communication of very *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

NEW How to Construct PRGs?



NEW PRG Length Extension

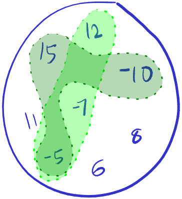


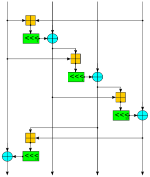
© ANSI/Archive.org

Plan for Today's Lecture…

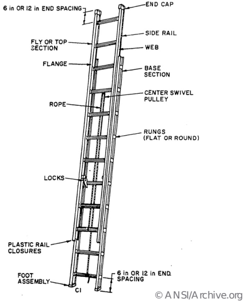
- Task: secure communication of very *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

NEW **How to Construct PRGs?**

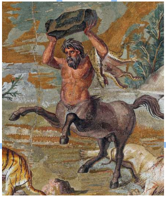




NEW **PRG Length Extension**



© ANSI/Archive.org



© Wikipedia

New tool: hybrid argument

How to Construct PRGs?

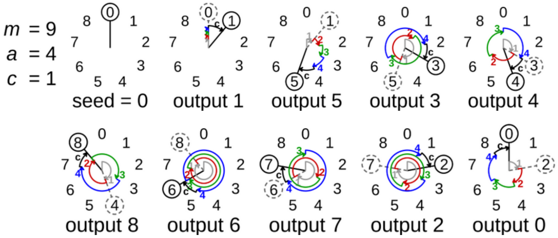
- Why not use Linear Congruential Generator (LCG)?
  - Used to generate randomness for simulating physical systems

How to Construct PRGs?

- Why not use Linear Congruential Generator (LCG)?
  - Used to generate randomness for simulating physical systems
  - Defined by recurrence relation $x_{n+1} = ax_n + c \bmod m$, with "seed" $x_0 \in [0, m-1]$

How to Construct PRGs?

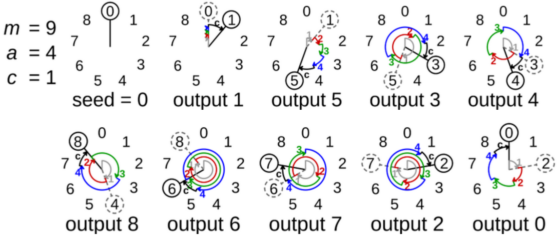
- Why not use Linear Congruential Generator (LCG)?
  - Used to generate randomness for simulating physical systems
  - Defined by recurrence relation $x_{n+1} = ax_n + c \bmod m$, with “seed” $x_0 \in [0, m-1]$



© Cmglee/Wikipedia

How to Construct PRGs?

- Why not use Linear Congruential Generator (LCG)?
  - Used to generate randomness for simulating physical systems
  - Defined by recurrence relation $x_{n+1} = ax_n + c \bmod m$, with “seed” $x_0 \in [0, m-1]$

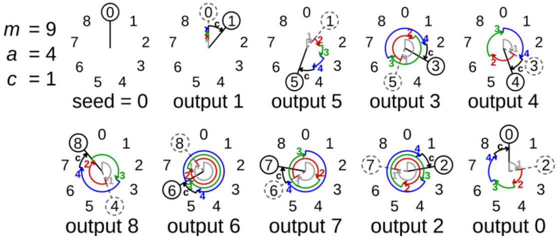


© Cmglee/Wikipedia

- Define PRG as $\mathsf{G}_{a,c}(s) := x_1 \| x_2$ with $x_0 := s$
  - How do you break $\mathsf{G}_{a,c}$?

How to Construct PRGs?

- Why not use Linear Congruential Generator (LCG)?
  - Used to generate randomness for simulating physical systems
  - Defined by recurrence relation $x_{n+1} = ax_n + c \bmod m$, with “seed” $x_0 \in [0, m-1]$



© Cmglee/Wikipedia

- Define PRG as $G_{a,c}(s) := x_1 \| x_2$ with $x_0 := s$
  - ❓ How do you break $G_{a,c}$?

⚠ Insecure for cryptographic purposes: “non-cryptographic” PRG!

How to Construct Cryptographic PRGs?...

Theoretical constructions

- *Direct constructions* from well-studied *hard problems*

How to Construct Cryptographic PRGs?...

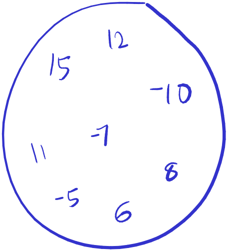
Theoretical constructions

- *Direct constructions* from well-studied *hard problems*
- E.g.: subset-sum problem:
  - Input: prime m and numbers $a_1, \ldots, a_n \in \mathbb{Z}_m$
  - Solution: $I \subseteq [1, n] : \sum_{i \in I} a_i = 0 \bmod m$

How to Construct Cryptographic PRGs?...

Theoretical constructions

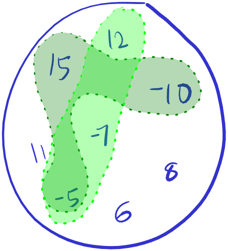
- *Direct constructions* from well-studied *hard problems*
- E.g.: subset-sum problem:
  - Input: prime m and numbers $a_1, \ldots, a_n \in \mathbb{Z}_m$
  - Solution: $I \subseteq [1, n] : \sum_{i \in I} a_i = 0 \bmod m$



How to Construct Cryptographic PRGs?...

Theoretical constructions

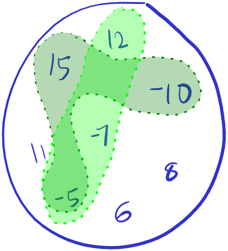
- *Direct constructions* from well-studied *hard problems*
- E.g.: subset-sum problem:
  - Input: prime m and numbers $a_1, \ldots, a_n \in \mathbb{Z}_m$
  - Solution: $I \subseteq [1, n] : \sum_{i \in I} a_i = 0 \bmod m$



How to Construct Cryptographic PRGs?

Theoretical constructions

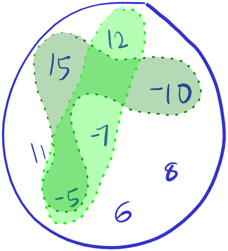
- *Direct constructions* from well-studied *hard problems*
- E.g.: subset-sum problem:
  - Input: prime m and numbers $a_1, \ldots, a_n \in \mathbb{Z}_m$
  - Solution: $I \subseteq [1, n] : \sum_{i \in I} a_i = 0 \bmod m$
- Believed to be “hard” (even for $a_1, \ldots, a_n \leftarrow \mathbb{Z}_m$)



How to Construct Cryptographic PRGs? ...

Theoretical constructions

- *Direct constructions* from well-studied *hard problems*
- E.g.: subset-sum problem:
  - Input: prime m and numbers $a_1, \dots, a_n \in \mathbb{Z}_m$
  - Solution: $I \subseteq [1, n] : \sum_{i \in I} a_i = 0 \bmod m$
- Believed to be "hard" (even for $a_1, \dots, a_n \leftarrow \mathbb{Z}_m$)



- PRG from subset-sum problem:

$$\mathsf{G}_{a_1,\dots,a_n}(x_1 \| \dots \| x_n) := \sum_{i \in [1,n]} x_i a_i \bmod m$$

  - Select $p \approx n^2 \Rightarrow G$ is expanding
  - Subset-sum problem hard $\Rightarrow \mathsf{G}_{a_1,\dots,a_n}$ pseudorandom

How to Construct Cryptographic PRGs?...

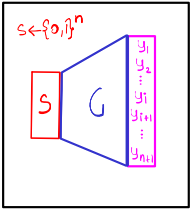
Theoretical constructions

- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)

How to Construct Cryptographic PRGs? ...

Theoretical constructions

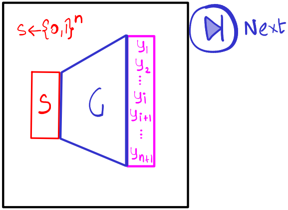
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs?

Theoretical constructions

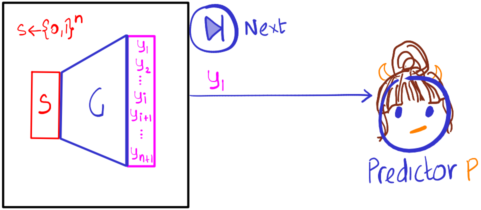
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs?...

Theoretical constructions

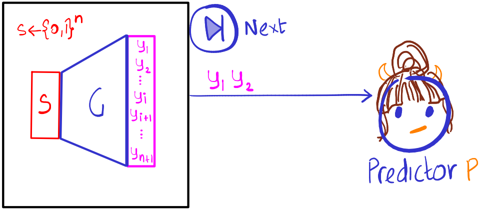
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs?...

Theoretical constructions

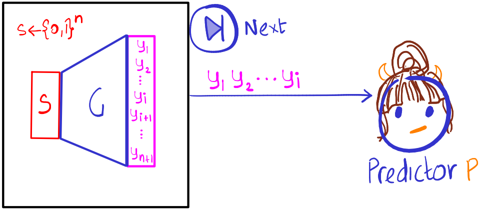
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs?...

Theoretical constructions

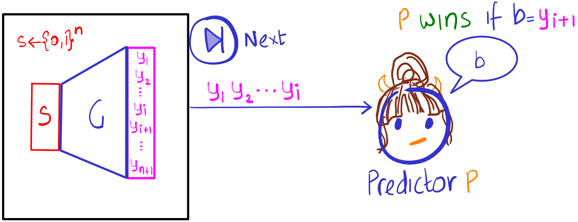
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs? …

Theoretical constructions

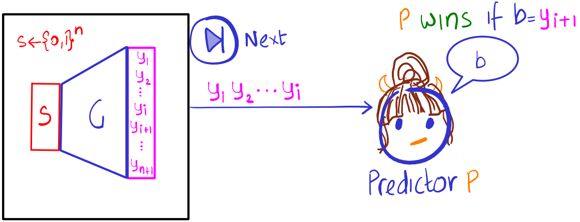
- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



How to Construct Cryptographic PRGs? ...

Theoretical constructions

- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



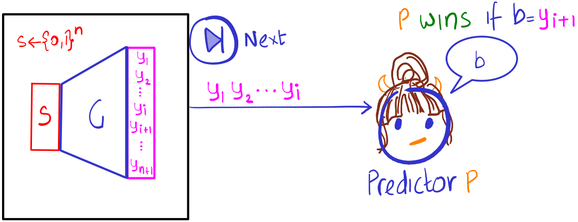
- E.g., Blum-Blum-Shub (BBS) sequence
  - Setting: modulus $m = pq$ for large primes p and q, seed $x \in \mathbb{Z}_m$
  - Sequence (modulo m):

$$LSB(x^2) \to LSB(x^{2^2}) \to LSB(x^{2^3}) \to \cdots \to LSB(x^{2^\ell}) \cdots$$

How to Construct Cryptographic PRGs?

Theoretical constructions

- Via *unpredictable sequences*: no PPT *predictor*, given a prefix of the sequence, can predict its next bit (non-negligibly away from 1/2)



- E.g., Blum-Blum-Shub (BBS) sequence
  - Setting: modulus $m = pq$ for large primes p and q, seed $x \in \mathbb{Z}_m$
  - Sequence (modulo m):

$$LSB(x^2) \rightarrow LSB(x^{2^2}) \rightarrow LSB(x^{2^3}) \rightarrow \cdots \rightarrow LSB(x^{2^\ell}) \cdots$$

  - Factoring m hard ⇒ sequence unpredictable
  - How to construct PRG from BBS sequence?

Do Cryptographic PRGs Exist?...

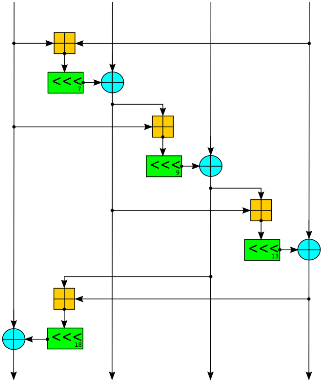
Practical constructions

- "Complex" functions, repeated "many times" look random
- Build a candidate construction and do extensive cryptanalysis
- E.g., Stream ciphers like Salsa20 and ChaCha

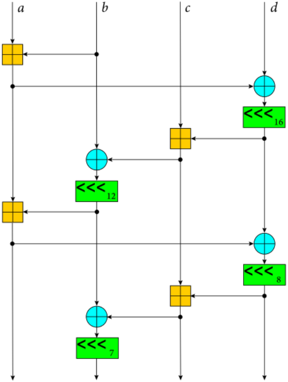
Do Cryptographic PRGs Exist?...

Practical constructions

- "Complex" functions, repeated "many times" look random
- Build a candidate construction and do extensive cryptanalysis
- E.g., Stream ciphers like Salsa20 and ChaCha



© Sissssou/Wikipedia

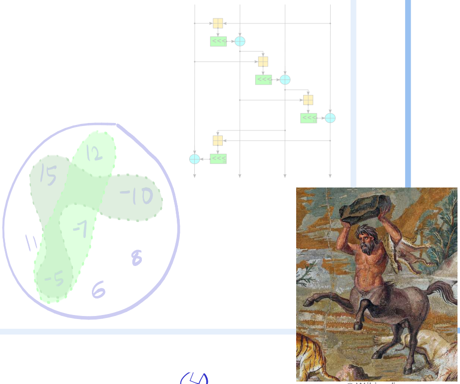


© Tony Arcieri/Wikipedia

Plan for Today's Lecture...

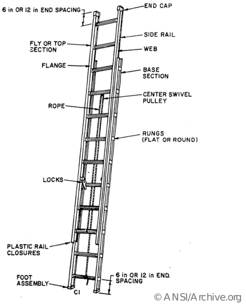
- Task: secure communication of very *long messages* with shared keys
- Threat model: computational secrecy against eavesdroppers

NEW How to Construct PRGs?



© Wikipedia

NEW PRG Length Extension



© ANSI/Archive.org

New tool: hybrid argument

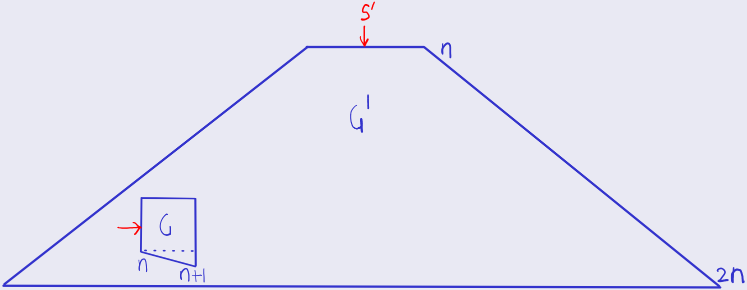
Let's Stretch!

- Goal: PRG G with stretch $n+1 \rightarrow$ PRG G$'$ with stretch $2n$

Let's Stretch!

- Goal: PRG G with stretch $n + 1 \rightarrow$ PRG G′ with stretch $2n$

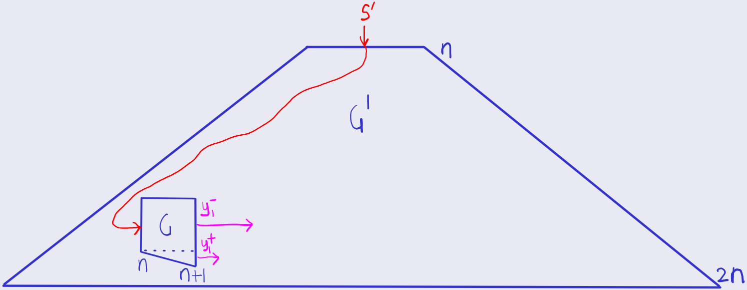
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1 \rightarrow$ PRG G′ with stretch $2n$

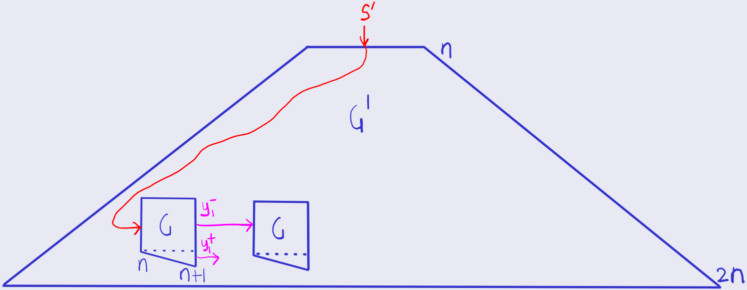
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1 \rightarrow$ PRG G′ with stretch $2n$

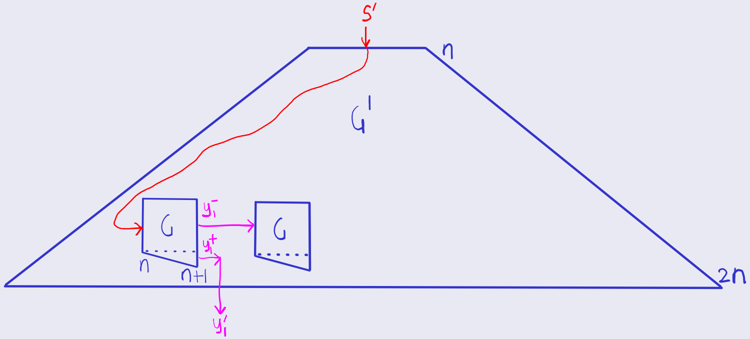
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1 \rightarrow$ PRG G$'$ with stretch $2n$

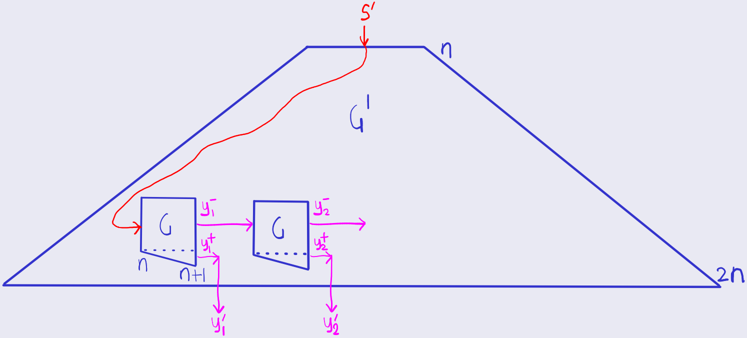
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1$ → PRG G′ with stretch $2n$

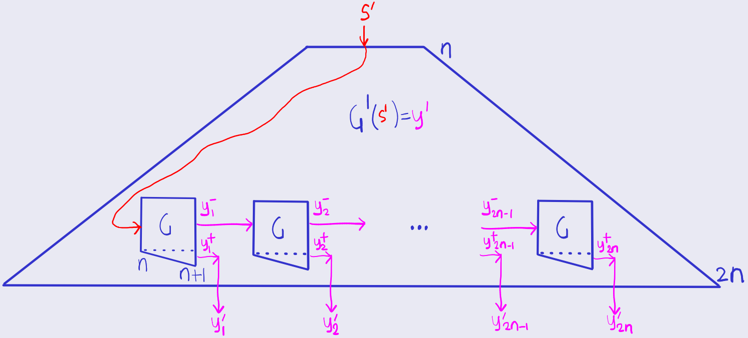
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1$ → PRG G′ with stretch $2n$

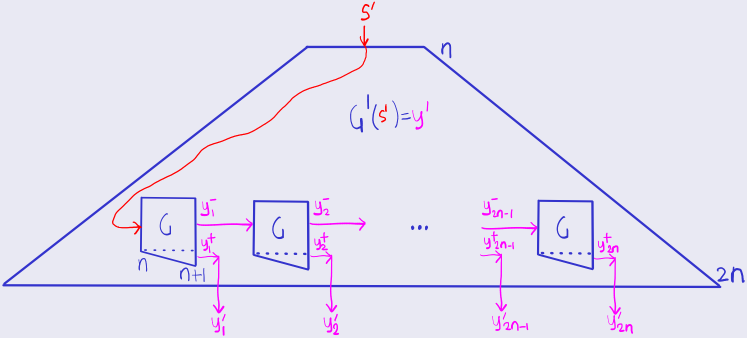
Construction 1



Let's Stretch!

- Goal: PRG G with stretch $n+1 \rightarrow$ PRG G′ with stretch $2n$

Construction 1



Exercise 1

Formally write down the construction of G′.

Before the Proof, Recall Definition of PRG Again

Definition 1 (Two-worlds definition)

Let G be an efficient deterministic algorithm that for any $n \in \mathbb{N}$ and input $s \in \{0,1\}^n$, outputs a string of length $\ell(n) > n$. ← "stretch/ expansion factor"
G is PRG if for every PPT *distinguisher* D

$$\delta(n) := \left| \Pr_{s \leftarrow \{0,1\}^n}[\mathsf{D}(G(s)) = 0] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[\mathsf{D}(r) = 0] \right|$$

↑ pseudorandom world ↑ random world

is negligible.

Before the Proof, Recall Definition of PRG Again

Definition 1 (Two-worlds definition)

Let G be an efficient deterministic algorithm that for any $n \in \mathbb{N}$ and input $s \in \{0,1\}^n$, outputs a string of length $\ell(n) > n$. ← "stretch/ expansion factor"

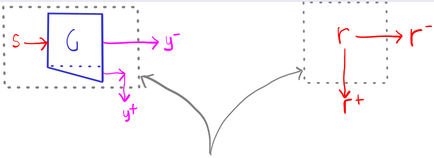
G is PRG if for every PPT *distinguisher* D

$$\delta(n) := \left| \Pr_{s \leftarrow \{0,1\}^n}[D(G(s)) = 0] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}[D(r) = 0] \right|$$

pseudorandom world

random world

is negligible.



For G stretching by one bit

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G'.

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof.

.

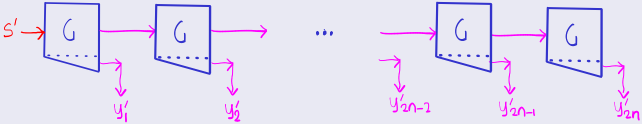
□

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. Intuition



□

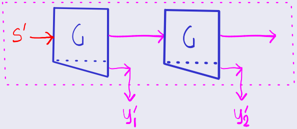
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. Intuition.

Let's focus on just two iterations



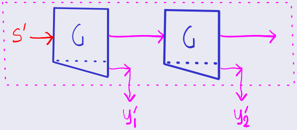
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

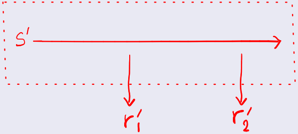
If G is a PRG, then so is G′.

Proof. Intuition .

Let's focus on just two iterations



H_0 pseudorandom world



H_2 random world

□

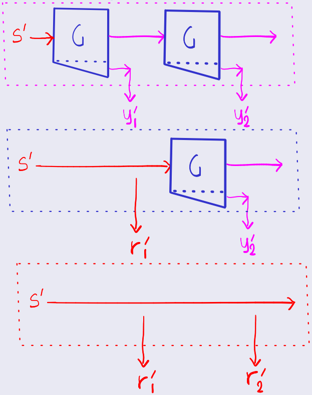
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

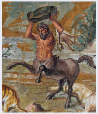
Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



H_0 pseudorandom world

H_1 hybrid world



H_2 random world

☐

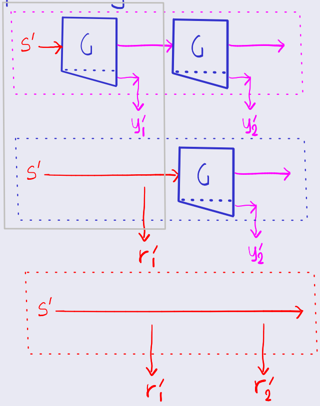
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

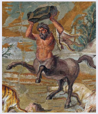
Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



H_0 pseudorandom world

H_1 hybrid world



H_2 random world

□

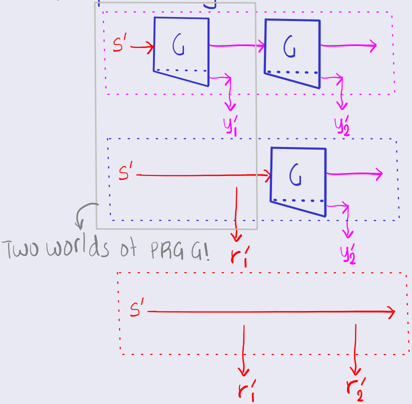
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

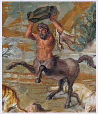
Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



H_0 pseudorandom world

H_1 hybrid world



H_2 random world

□

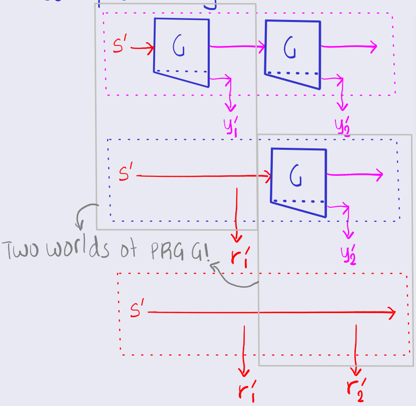
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

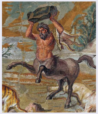
Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



H_0 pseudorandom world

H_1 hybrid world



H_2 random world

□

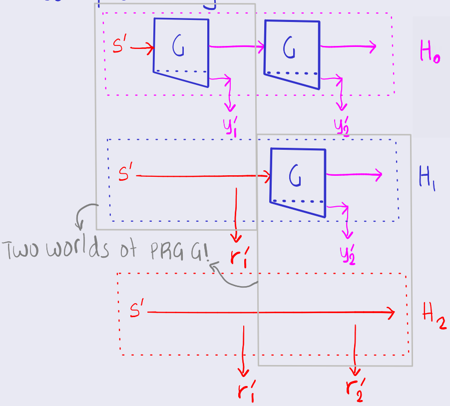
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



Claim 1

Distinguisher for G′

$$\Pr[D'(H_0)=0]-\Pr[D'(H_2)=0]\geq\delta$$

$$\Downarrow$$

$\exists\, i\in[0,1]$ such that

$$\Pr[D'(H_i)=0]-\Pr[D'(H_{i+1})=0]\geq\delta/2$$

□

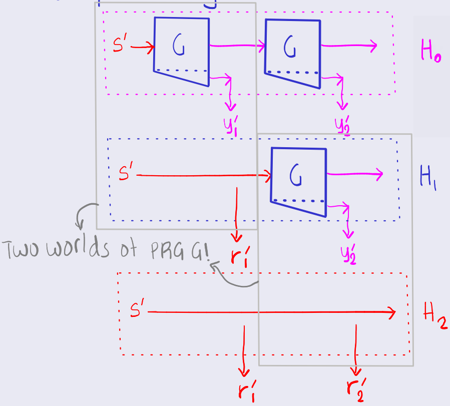
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



Claim 1

Distinguisher for G′

$$\Pr[D'(H_0)=0]-\Pr[D'(H_2)=0]\geq\delta$$

$$\Downarrow$$

$\exists\, i\in[0,1]$ such that

$$\Pr[D'(H_i)=0]-\Pr[D'(H_{i+1})=0]\geq\delta/2$$

☐

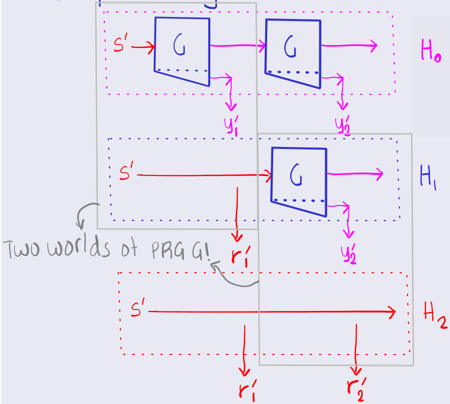
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



Claim 1

Distinguisher for G′

$$\Pr[D'(H_0)=0]-\Pr[D'(H_2)=0] \geq \delta$$

$$\Downarrow$$

$\exists i \in [0,1]$ such that

$$\Pr[D'(H_i)=0]-\Pr[D'(H_{i+1})=0] \geq \delta/2 \quad (*)$$

Claim 2

$(*) \Rightarrow$ Distinguisher D for G !

☐

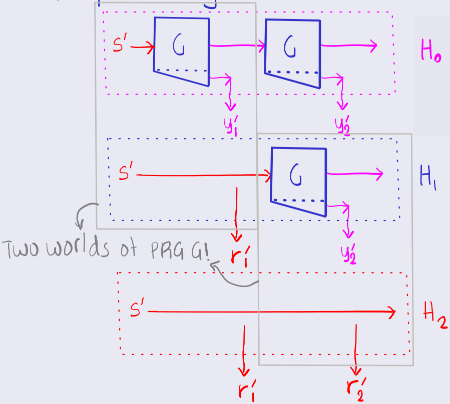
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

Let's focus on just two iterations



Claim 1

Distinguisher for G′

$$\Pr[D'(H_0)=0]-\Pr[D'(H_2)=0] \geq \delta$$

$$\Downarrow$$

$\exists\, i \in [0,1]$ such that

$$\Pr[D'(H_i)=0]-\Pr[D'(H_{i+1})=0] \geq \delta/2 \quad (*)$$

Claim 2

(*) ⇒ Distinguisher D for G !

Claims 1 & 2 ⇒ Theorem 1

□

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

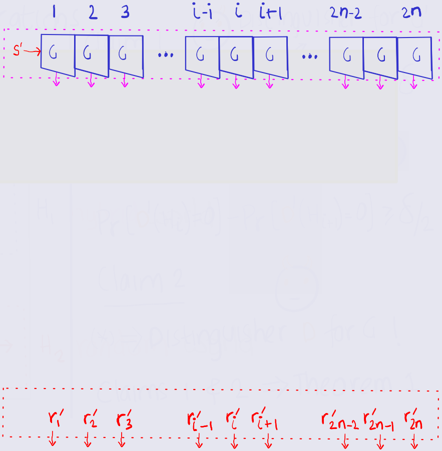
Proof. Intuition: consider hybrid worlds .

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G'.

Proof. Intuition: consider hybrid worlds .



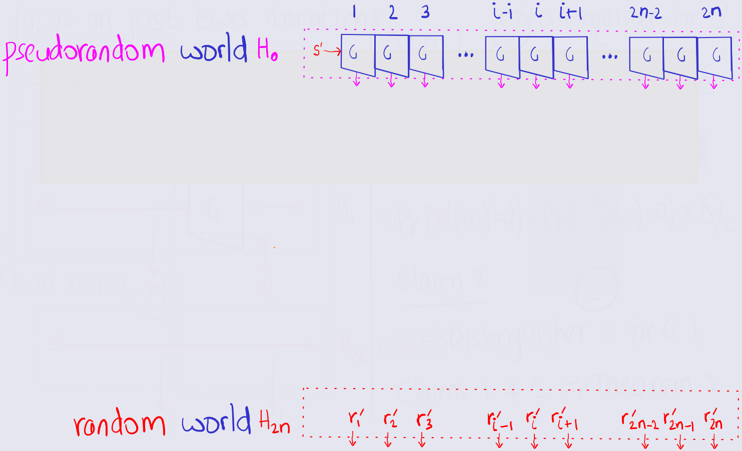
∎

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

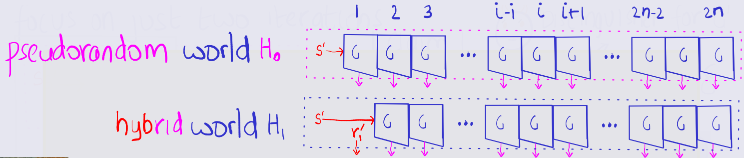


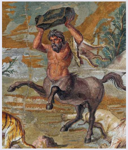
Proving Pseudorandomness: a Hybrid (Security) Argument

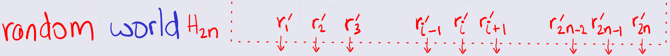
Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .







Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

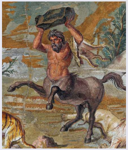
Proof. Intuition: consider hybrid worlds .

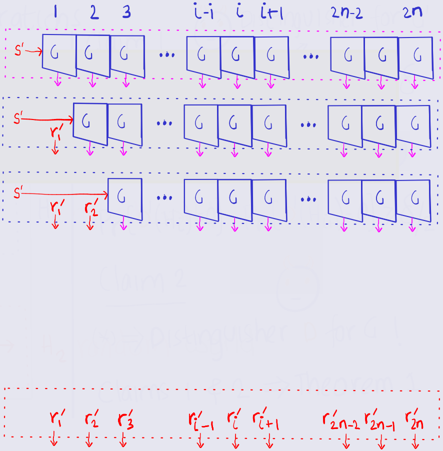
pseudorandom world H_0

hybrid world H_1

hybrid world H_2

random world H_{2n}



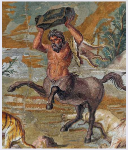


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .



pseudorandom world H_0

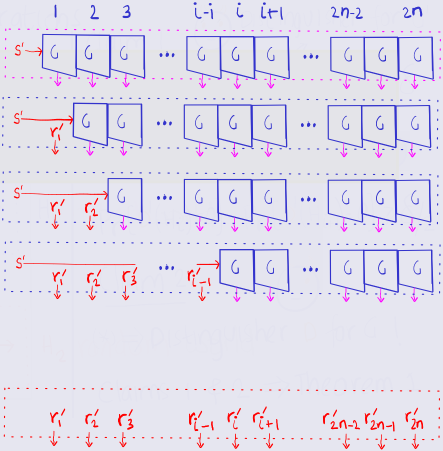
hybrid world H_1

hybrid world H_2

⋮

hybrid world H_{i-1}

random world H_{2n}



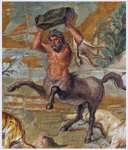
□

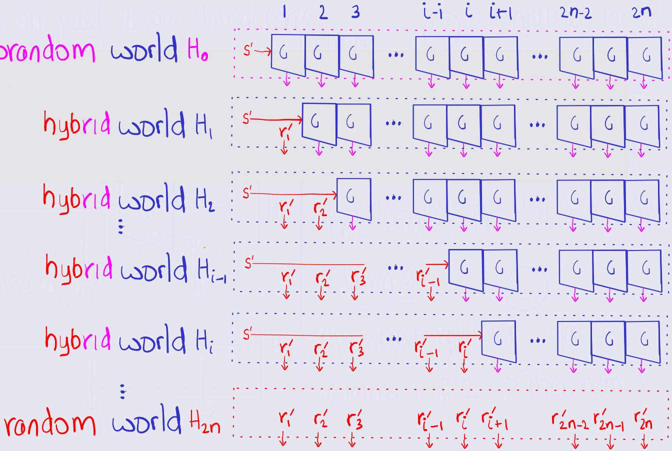
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. Intuition: consider hybrid worlds .



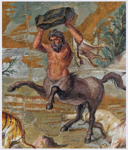


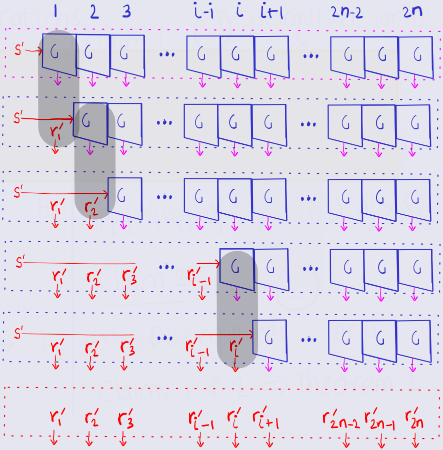
Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .





Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. Intuition: consider hybrid worlds .

Two worlds of PRG G′!

pseudorandom world H_0

hybrid world H_1

hybrid world H_2

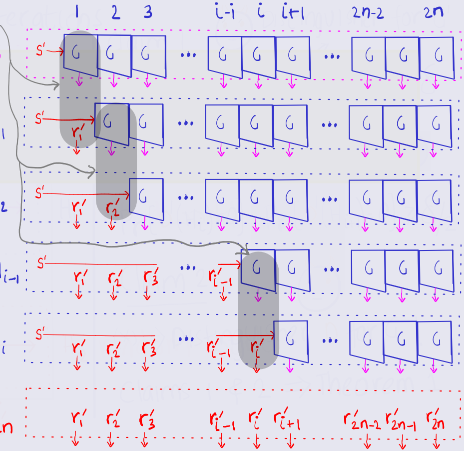
⋮

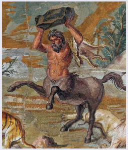
hybrid world H_{i-1}

hybrid world H_i

⋮

random world H_{2n}





∎

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G'.

Proof. $\exists$ distinguisher D for G $\Leftarrow$ $\exists$ distinguisher D' for G'.

□

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′*.*

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

PRG G′

Proving Pseudorandomness: a Hybrid (Security) Argument

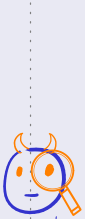
Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

PRG G

PRG G′



Distinguisher D

“Reduction”

Distinguisher D′

Proving Pseudorandomness: a Hybrid (Security) Argument

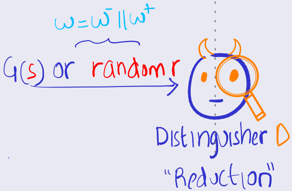
Theorem 2

If G is a PRG, then so is G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

PRG G

PRG G′



Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

PRG G

PRG G′

$G(s)\ \overbrace{\text{or random}}^{w = w^{-} \| w^{+}}\ r \longrightarrow$

Distinguisher D

"Reduction"

?

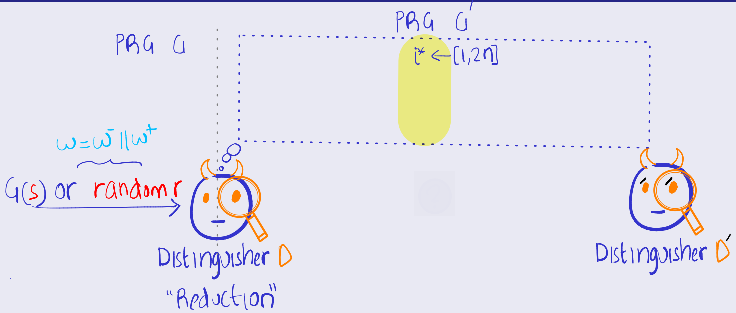
Distinguisher D′

Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

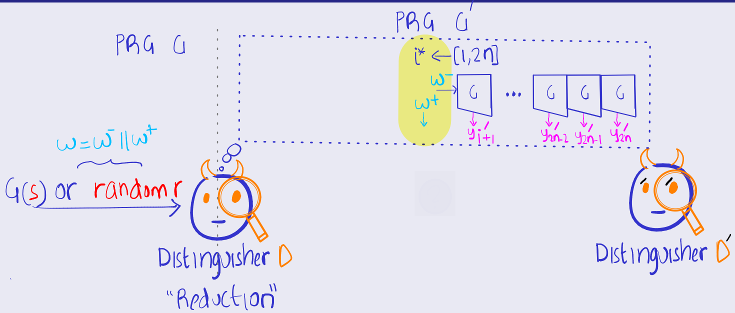


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

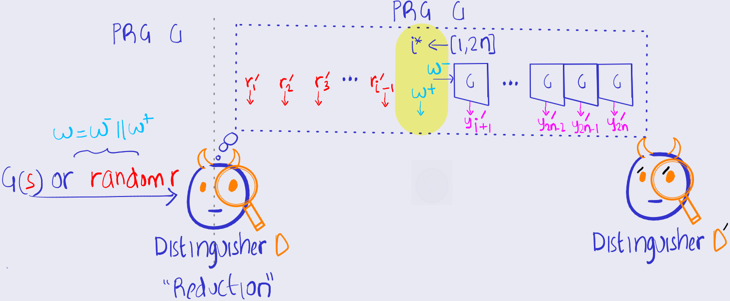


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G is a PRG, then so is G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

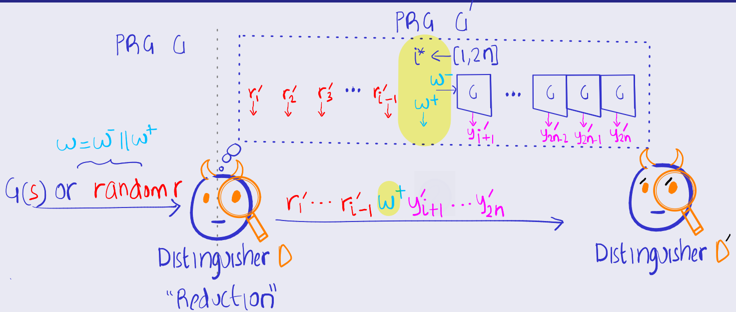


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

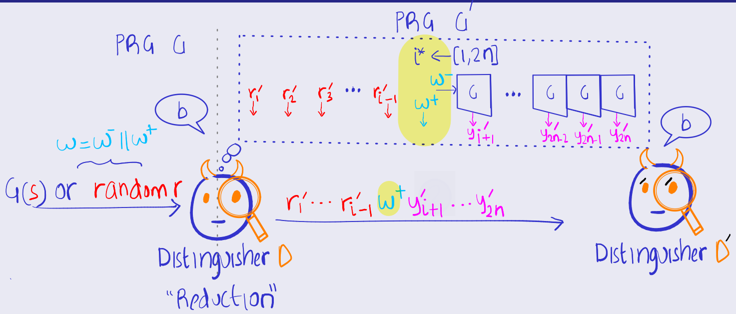


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.

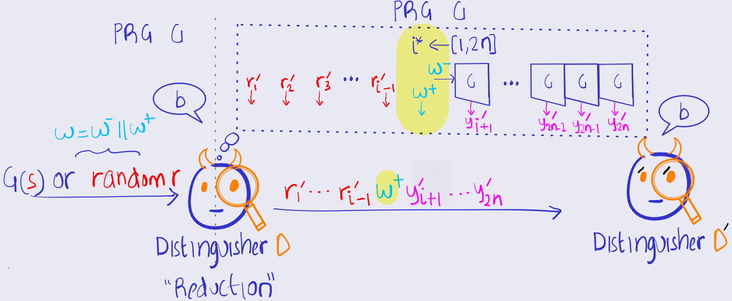


Proving Pseudorandomness: a Hybrid (Security) Argument

Theorem 2

If G *is a PRG, then so is* G′.

Proof. ∃ distinguisher D for G ⇐ ∃ distinguisher D′ for G′.



Analysis: similar to claims 1 and 2.

$\Pr[D'(H_0)=0] - \Pr[D'(H_{2n})=0] \geq \delta \Rightarrow$

$\Pr[i^*=i] = 1/2n$

$\exists i \in [0,2n-1]$ such that $\Pr[D'(H_i)=0] - \Pr[D'(H_{i+1})=0] \geq \delta/2n$

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch
  - ? What happens if we stretch it *exponentially*?

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch

  ? What happens if we stretch it *exponentially*?

- There is also a "loss in pseudorandomness"
  - D′ distinguishes with some probability $1/p(n) \Rightarrow$ D distinguishes with probability only $\approx 2n/p(n)$

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch
  - ? What happens if we stretch it *exponentially*?

- There is also a "loss in pseudorandomness"
  - D′ distinguishes with some probability $1/p(n) \Rightarrow$ D distinguishes with probability only $\approx 2n/p(n)$
  - More the stretch, greater the loss

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch
  - ? What happens if we stretch it *exponentially*?

- There is also a "loss in pseudorandomness"
  - D′ distinguishes with some probability $1/p(n) \Rightarrow$ D distinguishes with probability only $\approx 2n/p(n)$
  - More the stretch, greater the loss

- More generally: "loss in security" of a security reduction
  - One way to measure how "wasteful" the reduction is

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch
  - What happens if we stretch it *exponentially*?

- There is also a "loss in pseudorandomness"
  - D′ distinguishes with some probability $1/p(n) \Rightarrow$ D distinguishes with probability only $\approx 2n/p(n)$
  - More the stretch, greater the loss

- More generally: "loss in security" of a security reduction
  - One way to measure how "wasteful" the reduction is

Exercise 2

- Think of a less wasteful reduction strategy for Theorem 2. Do you feel it is possible?

Let's Take Stock of Theorem 2

- Construction 1 and Theorem 2 work for any polynomial stretch
  - ? What happens if we stretch it *exponentially*?

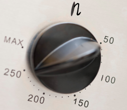
- There is also a "loss in pseudorandomness"
  - D′ distinguishes with some probability $1/p(n) \Rightarrow$
    D distinguishes with probability only $\approx 2n/p(n)$
  - More the stretch, greater the loss

- More generally: "loss in security" of a security reduction
  - One way to measure how "wasteful" the reduction is

Exercise 2

- Think of a less wasteful reduction strategy for Theorem 2. Do you feel it is possible?
- Maybe need a different construction?

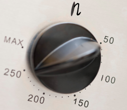
Why Loss in Security Matters?...



©freeimageslive.co.uk

- Suppose A running in n^3mins can solve a hard problem with probability $2^{40}/2^n$

Why Loss in Security Matters?...



©freeimageslive.co.uk

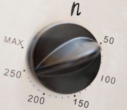
- Suppose A running in n^3mins can solve a hard problem with probability $2^{40}/2^n$

- What n do you choose while designing your scheme?

  No loss in security

1. $n = 50$?
   - A working for $\approx$ 3 months
   - Breaks with pr. $\approx 1/1000$
   - Acceptable
2. $n = 100$?
   - A working for $\approx$ 2years
   - Breaks with pr. 2^{-60}
   - Safe

Why Loss in Security Matters?...



©freeimageslive.co.uk

- Suppose A running in n^3mins can solve a hard problem with probability $2^{40}/2^n$
- What n do you choose while designing your scheme?

No loss in security

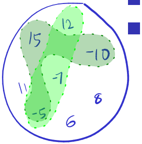
1. $n = 50$?
   - A working for $\approx$ 3 months
   - Breaks with pr. $\approx 1/1000$
   - Acceptable
2. $n = 100$?
   - A working for $\approx$ 2years
   - Breaks with pr. 2^{-60}
   - Safe

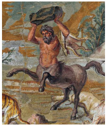
n loss in security

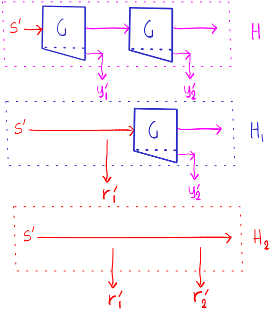
1. $n = 50$?
   - A working for $\approx$ 3 months
   - Breaks with pr. $\approx 1/20$
   - Breakable!
2. $n = 100$?
   - A working for $\approx$ 2years
   - Breaks with pr. $\approx 2^{-50}$
   - Safe

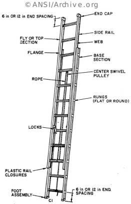
Recap/Next Lecture

- To recap:
  - Saw constructions of PRG
  - Increased the stretch of PRG
    - New tool: hybrid argument



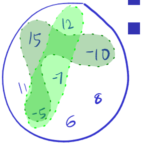


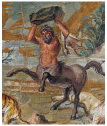


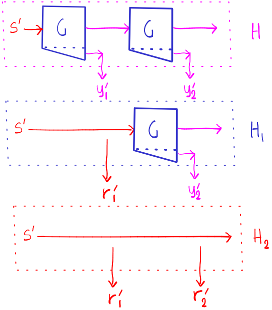

© ANSI/Archive.org

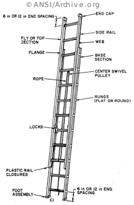
Recap/Next Lecture . . .

- To recap:
  - Saw constructions of PRG
  - Increased the stretch of PRG
    - New tool: hybrid argument









- Next lecture: How to encrypt *arbitrary-many* messages?
  - New primitive: pseudo-random function (PRF)
  - PRG → PRF (Goldreich-Goldwasser-Micali)
  - Stronger attack model: chosen-plaintext attack (CPA)

More Questions?

Further Reading

1. §3.3.2 in [Gol01] for more details on length-extension of PRG
2. For more details on stream ciphers, refer to §3.6.1 in [KL14] or §4 in [BS23]
3. To read more about unpredictability vs. pseudorandomness, see §3.3.5 in [Gol01]

Dan Boneh and Victor Shoup.
A Graduate Course in Applied Cryptography, Version 0.6.
2023.

Oded Goldreich.
The Foundations of Cryptography - Volume 1: Basic Techniques.
Cambridge University Press, 2001.

Jonathan Katz and Yehuda Lindell.
Introduction to Modern Cryptography (3rd ed.).
Chapman and Hall/CRC, 2014.