CIS 5560

Cryptography
Lecture 4

Course website:

pratyushmishra.com/classes/cis-5560-s24/

Slides adapted from Dan Boneh and Vinod Vaikuntanathan

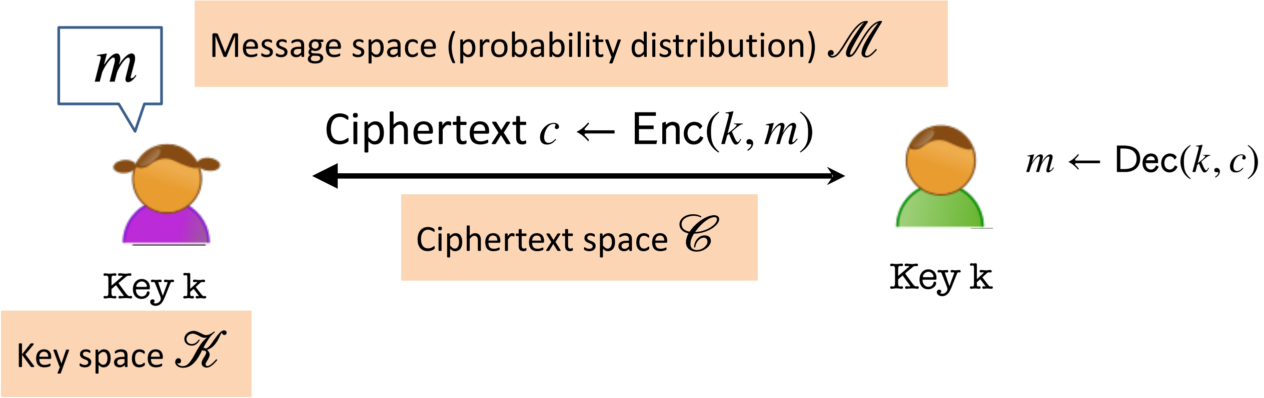
Announcements

- **HW 2 is out;** due Monday, Feb 5 at 5PM on Gradescope
  - Covers PRGs, OWFs, and semantic security
  - Get started today and make use of office hours!
- Cryptography related CIS Colloquium today after class
  - See what high level cryptography research looks like!
  - Bonus point on this week's homework if you attend!

Recap of last lecture

Key Notion: Secret-key Encryption

(or Symmetric-key Encryption)



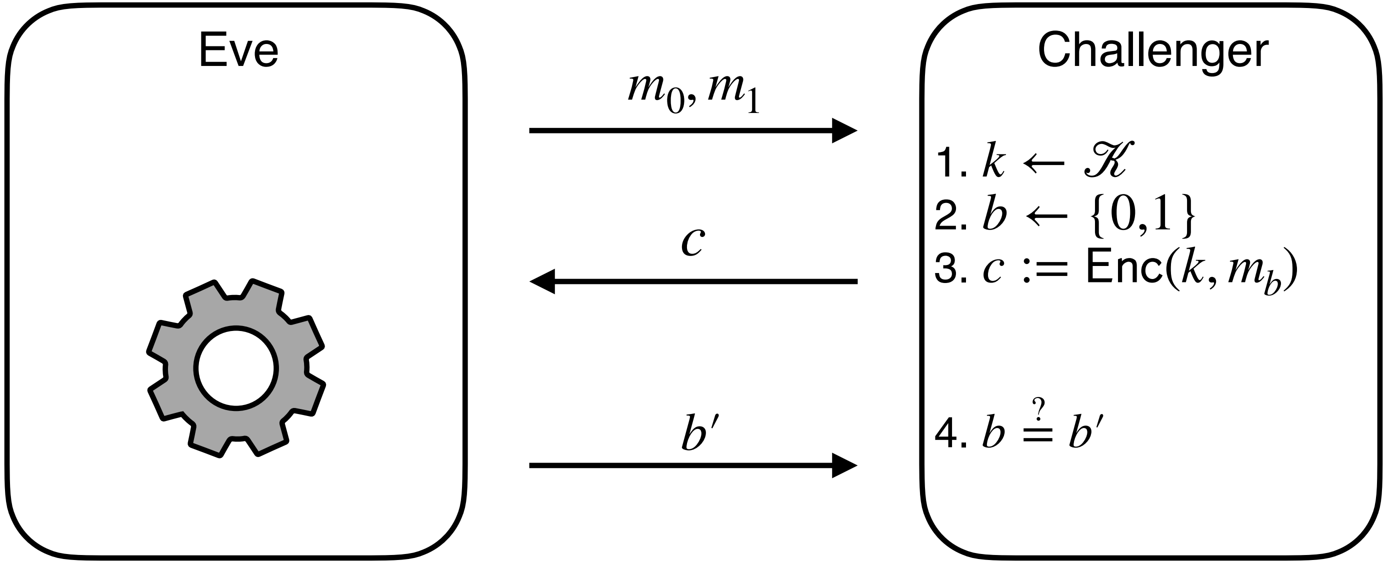
Three (possibly randomized) polynomial-time algorithms:

- **Key Generation Algorithm:** Gen$(1^k) \rightarrow k$

- **Encryption Algorithm:** Enc$(k, m) \rightarrow c$

- **Decryption Algorithm:** Dec$(k, c) \rightarrow m$

Semantic Security



Ans: we'll let Eve choose the messages!

PRG $\Longrightarrow$ Semantically Secure Encryption

(or, How to Encrypt n+1 bits using an n-bit key)

- Gen$(1^k) \to k$:
  - Sample an n-bit string at random.
- Enc$(k, m) \to c$:
  - Expand k to an $n+1$-bit string using PRG: $s = G(k)$
  - Output $c = s \oplus m$
- Dec$(k, c) \to m$:
  - Expand k to an $n+1$-bit string using PRG: $s = G(k)$
  - Output $m = s \oplus c$

Correctness:

$Dec(k, c)$ outputs $G(k) \oplus c = G(k) \oplus G(k) \oplus m = m$

Distinguisher $D(y)$:

1. Get two messages m_0, m_1, from Eve and sample a bit b
2. Compute $b' \leftarrow \text{Eve}(y \oplus m_b)$
3. If $b' = b$, output "PRG"
4. Otherwise, output "Random"

World 0

$\Pr[D \text{ outputs "PRG"} \mid y \text{ is pseudorandom}]$
$= \Pr[\text{Eve outputs } b' = b \mid y \text{ is pseudorandom}]$
$= \rho \geq 1/2 + 1/p(n)$

World 1

$\Pr[D \text{ outputs "PRG"} \mid y \text{ is random}]$
$= \Pr[\text{Eve outputs } b' = b \mid y \text{ is random}]$
$= \rho' = 1/2$

Therefore,

$$\left| \Pr[D \text{ outputs "PRG"} \mid y \text{ is pseudorandom}] - \Pr[D \text{ outputs "PRG"} \mid y \text{ is random}] \right|$$
$$\geq 1/p(n)$$

$\blacksquare$

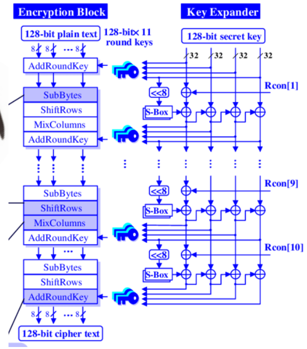
Constructing PRGs: Two Methodologies

The Practical Methodology

1. Start from a design framework
(e.g. “appropriately chosen functions composed appropriately many times look random”)

2. Come up with a candidate construction

3. Do extensive cryptanalysis.

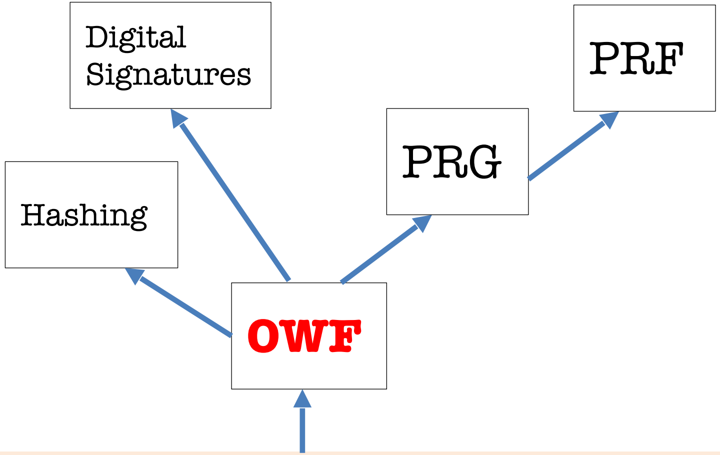


Constructing PRGs: Two Methodologies

<u>The Foundational Methodology (much of this course)</u>

Reduce to simpler primitives.

"Science wins either way" –Silvio Micali

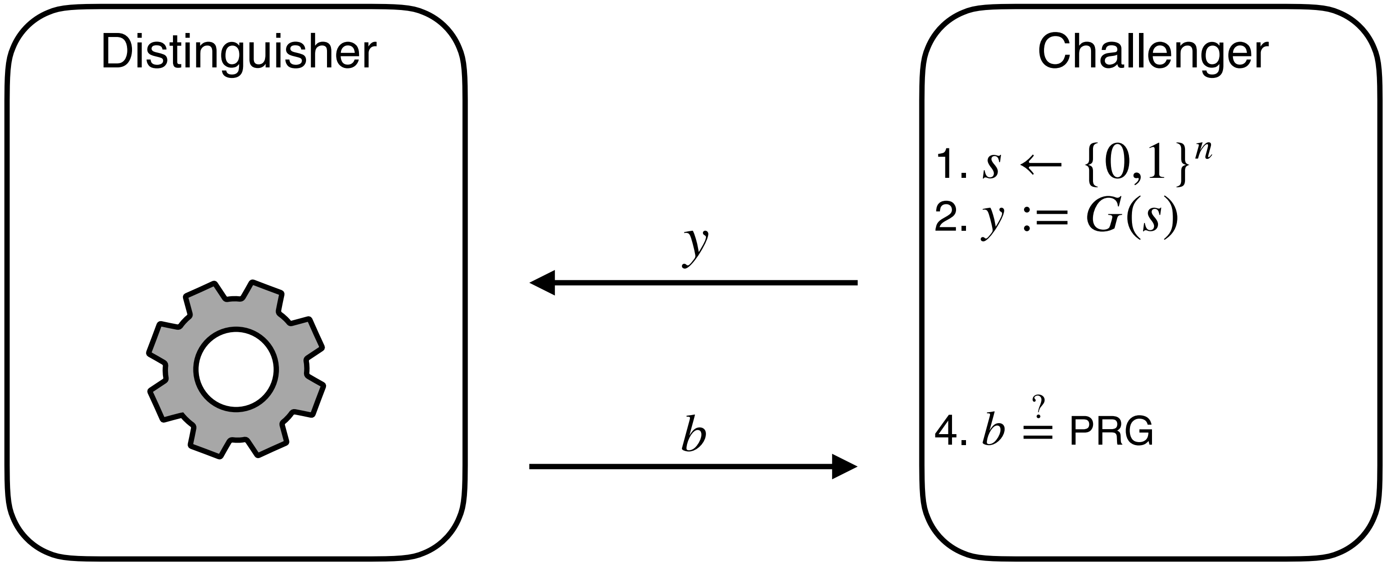


well-studied, average-case hard, problems

Today’s Lecture

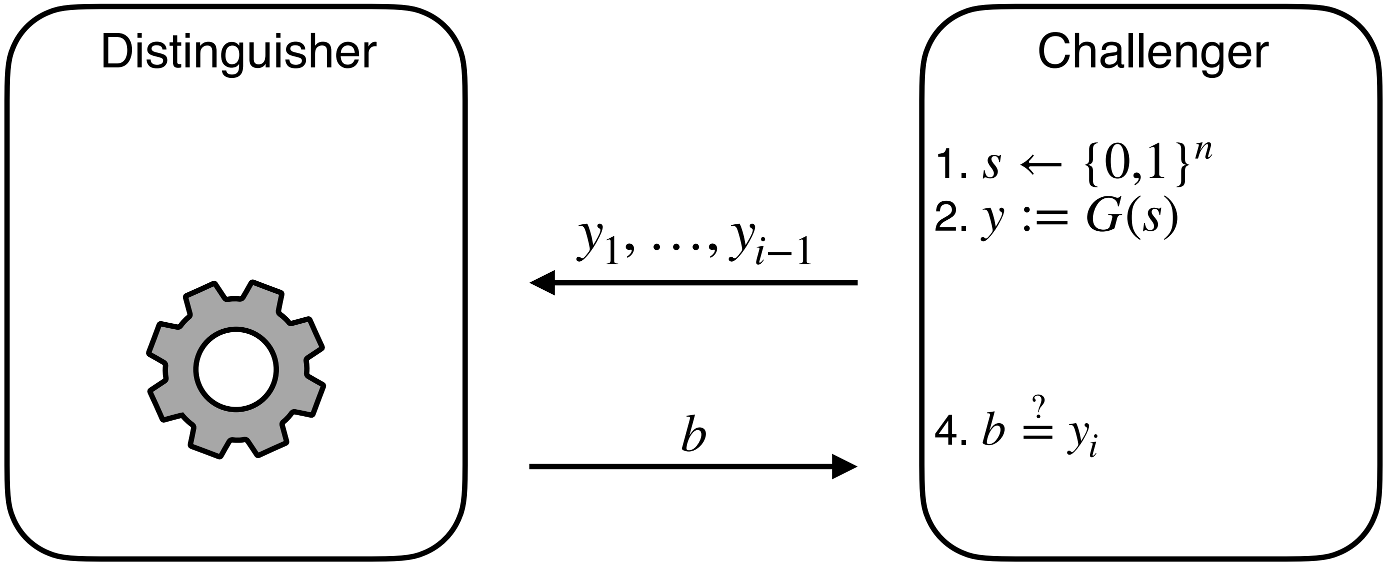
- PRG Indistinguishability → PRG Unpredictability
- One way functions and permutations
- OWPs → PRGs

PRG Indistinguishability



$$\left| \Pr[D(G(U_n)) = 1] - \Pr[D(U_m) = 1] \right| = \varepsilon(n)$$

PRG Next-Bit Unpredictability



$$\Pr\left[A(y_1, \ldots, y_{i-1}) = y_i \,\middle|\, \begin{array}{r} s \leftarrow \{0,1\}^n \\ y \leftarrow G(s) \end{array}\right] = 1/2 + \varepsilon(n)$$

PRG Def 2: Next-bit Unpredictability

Definition [Next-bit Unpredictability]:

A deterministic polynomial-time computable function G: $\{0,1\}^n \rightarrow \{0,1\}^m$ is next-bit unpredictable if:

for every PPT algorithm P (called a next-bit predictor) and every $i \in \{1,\ldots,m\}$, *if there is a negligible function* μ *such that:*

$$\Pr\left[y \leftarrow G(U_n) : P(y_1 y_2 \ldots y_{i-1}) = y_i\right] = \frac{1}{2} + \mu(n)$$

Notation: $y_1, y_2, \ldots y_m$ are the bits of the m-bit string y.

Def 1 and Def 2 are Equivalent

Theorem:

A PRG G is indistinguishable if and only if it is next-bit unpredictable.

Def 1 and Def 2 are Equivalent

Theorem:

A PRG G passes all PPT distinguishers if and only if it passes PPT *next-bit* distinguishers.

NBU and Indistinguishability

- Next-bit Unpredictability (NBU): Seemingly much weaker requirement. Only says that next bit predictors, a particular type of distinguishers, cannot succeed.

- Yet, surprisingly, Next-bit Unpredictability (NBU) = Indistinguishability.

- NBU often much easier to use.

1. Indistinguishability $\Longrightarrow$ NBU

Proof: by contradiction.

Suppose for contradiction that there is a p.p.t. predictor P, a polynomial function p and an $i \in \{1,\ldots,m\}$ s.t.

$$\Pr\left[\, y \leftarrow G(U_n) : P(y_1 y_2 \ldots y_{i-1}) = y_i \right] \geq \frac{1}{2} + 1/p(n)$$

Then, I claim that P essentially gives us a distinguisher D!

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.

2. If P returns the i-th bit y_i, then output 1 ("PRG") else output 0 ("Random").

If P is p.p.t. so is D.

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \dots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

We want to show: there is a polynomial p' s.t.

$$\begin{aligned} &| \Pr[y \leftarrow G(U_n) :\ D(y) \ = 1\ ] \\ &- \Pr[y \leftarrow Um :\ D(y) \ = 1\ ]\ | \geq 1/p'(n) \end{aligned}$$

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.

2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

$$\Pr[y \leftarrow G(U_n) : D(y) = 1]$$

$$= \Pr[y \leftarrow G(U_n) : P(y_1 y_2 \ldots y_{i-1}) = y_i] \quad \text{(by construction of D)}$$

$$\geq \frac{1}{2} + 1/p(n) \quad \text{(by assumption on P)}$$

1. Indistinguishability $\implies$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (= "PRG") else output 0 (= "Random").

$$\Pr[y \leftarrow G(U_n) : D(y) = 1] \geq \frac{1}{2} + 1/p(n)$$

$$\Pr\left[y \leftarrow U_m : D(y) = 1\right]$$

$$= \Pr[y \leftarrow U_m : P(y_1 y_2 \ldots y_{i-1}) = y_i]$$ (by construction of D)

$$= \frac{1}{2}$$ (since y is random)

1. Indistinguishability $\Longrightarrow$ NBU

Consider D which gets an m-bit string y and does the following:

1. Run P on the $(i-1)$-bit prefix $y_1 y_2 \ldots y_{i-1}$.
2. If P returns the i-th bit y_i, then output 1 (= “PRG”) else output 0 (= “Random”).

$$\Pr[y \leftarrow G(U_n) : D(y) = 1] \geq \frac{1}{2} + 1/p(n)$$

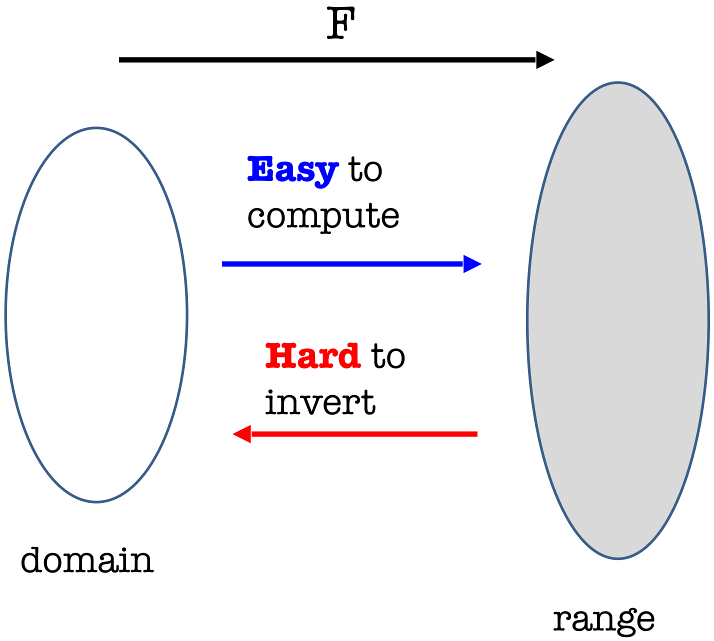
$$\Pr\left[y \leftarrow U_m : D(y) = 1\right] = \frac{1}{2}$$

So, $|\Pr[y \leftarrow G(U_n) : D(y) = 1] - \Pr[y \leftarrow Um : D(y) = 1]| \geq 1/p(n)$ $\blacksquare$

Today's Lecture

- PRG Indistinguishability → PRG Unpredictability
- **How to construct PRGs?**
  - **One way functions and permutations**
- OWPs → PRGs

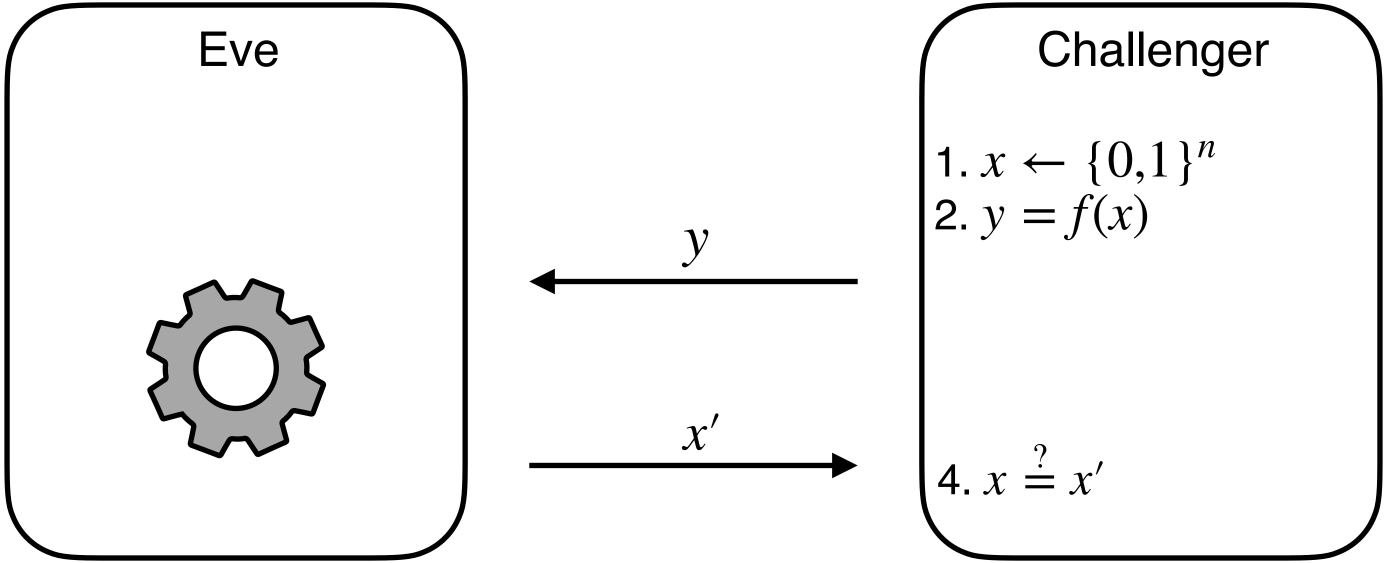
One-way Functions (Informally)



Source of all hard problems in cryptography!

What is a good definition?

OWF Security Attempt #1



One-way Functions (Take 1)

A function (family) $\{F_n\}_{n\in\mathbb{N}}$ where $F(\cdot):\{0,1\}^n \to \{0,1\}^{m(n)}$ is **one-way** if for every p.p.t. adversary A, the following holds:

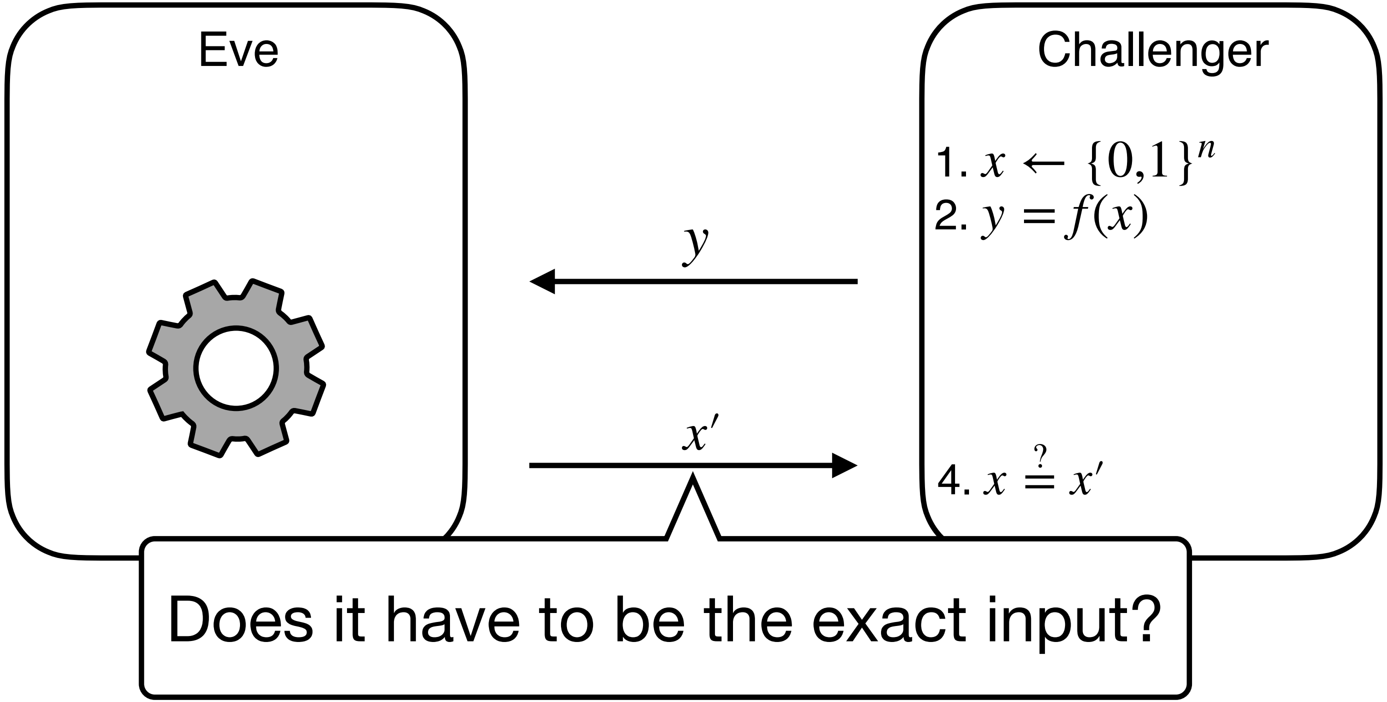
$$\Pr\left[A(1^n, y) = x \,\middle|\, \begin{array}{c} x \leftarrow \{0,1\}^n \\ y := F_n(x) \end{array}\right] = \text{negl}(n)$$

Consider $\boldsymbol{F_n(x) = 0}$ for all x.

This is one-way according to the above definition.
In fact, impossible to find *the* inverse even if A has unbounded time.

Conclusion: not a useful/meaningful definition.

OWF Security Attempt #2



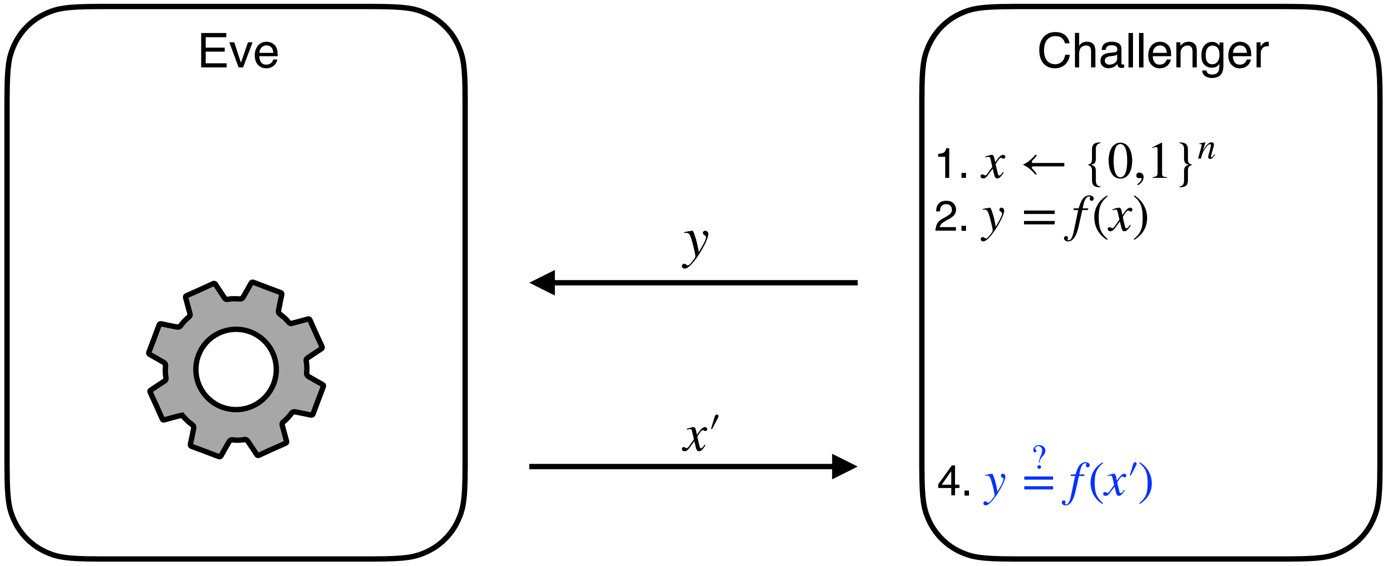
One-way Functions (Take 1)

A function (family) $\{F_n\}_{n\in\mathbb{N}}$ where $F(\cdot):\{0,1\}^n \to \{0,1\}^{m(n)}$ is **one-way** if for every p.p.t. adversary A, the following holds:

$$\Pr\left[A(1^n, y) = x \,\middle|\, \begin{array}{r} x \leftarrow \{0,1\}^n \\ y := F_n(x) \end{array}\right] = \mathrm{negl}(n)$$

The Right Definition: Impossible to find ***an*** inverse efficiently.

OWF Security Attempt #2



One-way Functions: The Definition

A function (family) $\{F_n\}_{n \in \mathbb{N}}$ where $F(\cdot) : \{0,1\}^n \to \{0,1\}^{m(n)}$ is **one-way** if for every p.p.t. adversary A, the following holds:

$$\Pr\left[F_n(x') = y \;\middle|\; \begin{array}{r} x \leftarrow \{0,1\}^n \\ y := F_n(x) \\ x' \leftarrow A(1^n, y) \end{array} \right] = \text{negl}(n)$$

- Can always find *an* inverse with unbounded time
- … but should be hard with probabilistic polynomial time

One-way Permutations:

One-to-one one-way functions with $m(n) = n$.

How to get PRG from OWF?

OWF → PRG, Attempt #1

PRG(k)

1. Output $F_n(k)$

(Assume $m(n) > n$)

Does this work?

OWF → PRG, Attempt #1

Consider $F_n(x)$ constructed from another OWF F'_n:

1. Compute $y := F'_n(x)$
2. Output $y' := (y_0,\ 1, y_1,\ 1, \ldots, y_n,\ 1)$

PRG(k)

1. Output $F_n(k)$

Is F one-way?

Yes!

Is PRG unpredictable?

No!

Our problem:

OWFs don’t tell us anything about how their outputs are distributed.

They are only hard to invert!

Hardcore Bits

If F is a one-way function, we know it's hard to compute a pre-image of $F(x)$ for a randomly chosen x.

How about computing partial information about an inverse?

Exercise: There are one-way functions for which it is easy to compute the first half of the bits of an inverse.

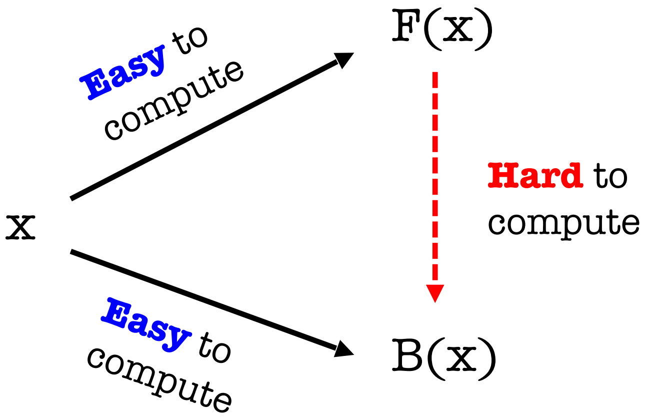
Hardcore Bits

HARDCORE PREDICATE (Definition)

For any function (family) $F: \{0,1\}^n \rightarrow \{0,1\}^m$, a function $B: \{0,1\}^n \rightarrow \{0,1\}$ is a hardcore **predicate** if for every p.p.t. adversary A, there is a negligible function μ s.t.

$$\Pr\left[x \leftarrow \{0,1\}^n; y = F(x): A(y) = B(x)\right] \leq \frac{1}{2} + \mu(n)$$

Hardcore Predicate (in pictures)



Next class

- How to get randomness from OWF output
  - How to use this to get PRGs
- How to extend the length of PRGs
- How to get PRGs with “exponentially-large” output