Review of last time

- Principle of provable security:
  - Define your model: syntax, adversary, goal
  - Design your protocol
  - Set your assumptions
  - Prove security

- Pseudorandom generators:
  - Obtain big randomness from small seed
  - Security: indistinguishability from random

Course objectives

- To understand:
  - The principles of *security proofs* ✓
  - Typical *security models* for various primitives ✓ ✗
  - Basic security *reductions* ✓
  - Necessary and *sufficient assumptions* for proofs
- To become able to:
  - Given a proof strategy, *compute success* of reduction
  - Design a *proof strategy* from scratch
  - Write *full basic proofs*
  - *Assess the soundness* of a proof, spot irregularities
  - *Draw conclusions* about impact of a proof

INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

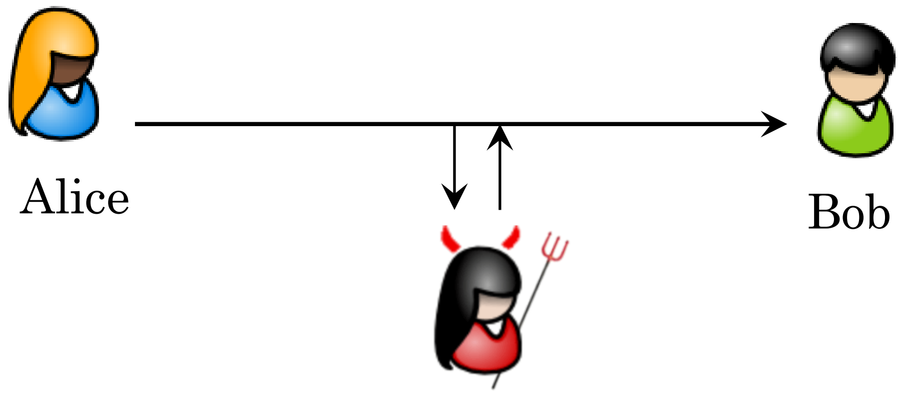
UMR IRISA

Symmetric Ciphers and PRGs

Symmetric Encryption, Perfect Ciphers, Definitions of PRGs and PRFs

Encryption Schemes

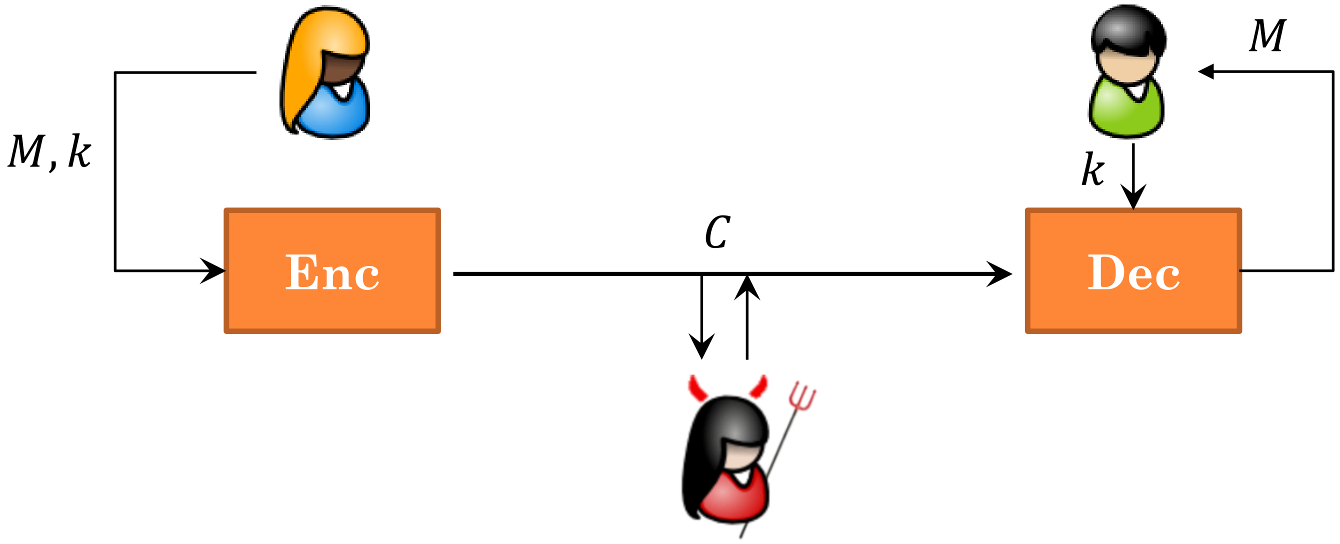
- Designed to protect message confidentiality
  - Usually 2 parties, called Alice and Bob; adversary is Eve
  - Plaintext M encrypted by Alice, becoming a ciphertext C
  - Ciphertext C decrypted by Bob to some plaintext M'
  - Necessary: Bob (and maybe Alice) must have a secret k



Secrets and Non-Secrets

- Kerckhoff: Consider the algorithm public
  - If the algorithm is compromised, no problem
  - More eyes to look at the security of a public algorithm

- Symmetric-key encryption (block/stream ciphers)
  - Alice and Bob share secret key k



Basic Ciphers

- Caesar cipher and extensions
  - Permutation cipher
  - Key is the number of letters we permute by
  - Caesar: $k = 3$
  - BLOCKCIPHER becomes EORFNFLSKHU

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U

S	T	U	V	W	X	Y	Z
V	W	X	Y	Z	A	B	C

The Caesar Cipher

- Kerckhoff: algorithm is public
- We need the key
  - Key space is too small : brute force works in one go with probability $\frac{1}{26}$ and works for sure in 26 attempts
  - Attack works only if message is meaningful

Brute force is base line for attacks against ciphers

ONE-TIME PAD

- Substitution cipher, C = M + K (e.g. mod 26)
- Key length equal to message length
- If M = BLOCKCIPHER, and K = PRZANIBQTCS
- Say message is meaningful and key is meaningful
  - Can we do better than brute force?

    Yes, look at language statistics
- Say message is meaningful, but key is truly random
  - Key hides message information-theoretically

B	L	O	C	K	C	I	P	H	E	R
P	R	Z	A	N	I	B	Q	T	C	S
R	C	N	D	Y	L	K	F	A	H	J

Security details

- What if same key used multiple times in N attempts?
  - Case 1: Adversary knows it (described in protocol)

    Passive eavesdropper learns M_1 XOR M_2

    Equivalent to using meaningful key
  - Case 2: Adversary does not know (accidental collision)

    Even assuming this is problematic,

    this happens rarely (w.p. $\leq \binom{N}{2} 2^{-|sk|}$)
- What does it mean that the key "hides" a message?
  - BLOCKCIPHER + "PRZANIBQTCS" = RCNDYLKFAHJ
    UNIVERSALLY + "XOEHUUSEPWO" = RCNDYLKFAHJ
    YETIMONSTER + "TYUVLXXNGCS" = RCNDYLKFAHJ
  - Message is meaningful: probability bound by dictionary attack

Guarantee of One-time Pad

- Ingredients:
  - Set $\mathcal{S}$, which is an alphabet (like A, B, ..., Z)
  - Length of messages l
  - Subset $\mathcal{M} \in \mathcal{S}^l$ of meaningful messages of length l
  - An (Abelian) group operation $" + "$ on $\mathcal{S}^l$, inverse operation "$-$"
- Guarantee:
  - The cipher consisting of:
    - Picking K randomly from $\mathcal{S}^l$
    - Encrypting plaintext $M \in \mathcal{M}$ to $C \ := M + K$
    - Decrypting plaintext C to $M = C - K$

    guarantees that:

$$\text{Prob}[\text{ptext} = M \mid \text{ctext} \ = C\ ] \ = \text{Prob}\ [\text{ptext} = M\ ]$$

Perfect cipher

Perfect ciphers

- Perfect ciphers:

$$\text{Prob}[\text{ptext} = M \mid \text{ctext} = C] = \text{Prob}[\text{ptext} = M]$$

  - Ciphertext gives no information on plaintext

- Theorem 1:
  - Take a perfect cipher with plaintext alphabet $\mathcal{M}$ (all messages occuring with non-zero probability) and key space $\mathcal{K}$
  - Then the size of $\mathcal{K}$ is at least equal to the size of $\mathcal{M}$

- Proof:
  - First observation: take plaintexts $M_1 \neq M_2$. Then for all $k \in \mathcal{K}$ it holds that $Enc(k; M_1) \neq Enc(k;\ M_2)$. Why?

Key-size of perfect ciphers

- Theorem 1:
  - Take a perfect cipher with plaintext alphabet $\mathcal{M}$ (all messages occuring with non-zero probability) and key space $\mathcal{K}$
  - Then the size of $\mathcal{K}$ is at least equal to the size of $\mathcal{M}$

- Proof:
  - Reduction to absurd: Suppose $|\mathcal{K}| \leq |\mathcal{M}| - 1$
  - Look at mapping $(M,\ k) \rightarrow\ C$ (through encryption)
    - Order $\mathcal{M}$ in some way (lexicographically or just randomly)
    - Take the first message, denote it M_1
    - Pick key k_1, compute $C =\ Enc(k_1, M_1)$. If $C =$ ב (invalid), pick again
    - Continue picking keys $k\ \neq\ k_1$ and run $Dec\ (C, k)$
  - Even if all decryptions give a valid result, Obs 1 tells us there exists at least one M^* that C does not decrypt to.

Key-size of perfect ciphers

- Theorem 1:
  - Take a perfect cipher with plaintext alphabet $\mathcal{M}$ (all messages occuring with non-zero probability) and key space $\mathcal{K}$
  - Then the size of $\mathcal{K}$ is at least equal to the size of $\mathcal{M}$

- Proof:
  - Reduction to absurd: Suppose $|\mathcal{K}| \leq |\mathcal{M}| - 1$
  - Look at mapping $(M, k) \rightarrow C$ (through encryption)
  - Even if all decryptions give a valid result, Obs 1 tells us there exists at least one M^* that C does not decrypt to
  - Then for this message it holds that:

$$\text{Prob}[\text{ptext} = M^* \mid \text{ctext} = C] = 0 \neq \text{Prob}[\text{ptext} = M_1 | \, ctext = C]$$

  - This is impossible (perfect cipher)
  - Hence $|\mathcal{K}| \geq |\mathcal{M}|$

INDISTINGUISHABILITY

- Consequence of Theorem 1:
  - OTP has optimal key size (and it's long!)

- Another way to phrase perfection property:
  - Indistinguishability:

    For any messages $M_1 \neq M_2$ and any ciphertext C :

    $$\mathrm{Prob}[\mathrm{Enc}(*, M_1) = C] = \mathrm{Prob}[\mathrm{Enc}(*, M_2) = C]$$

- Theorem 2: A cipher is perfect if, and only if, it has the indistinguishability property

  Proof: in the TDs.

Some Conclusions

- Perfect ciphers:
  - Ciphertext reveals nothing about the plaintext
  - Equivalently phrased as: each ciphertext could correspond to any plaintext
  - … But they require $|\mathcal{K}| \geq |\mathcal{M}|$
- One Time Pad (OTP):
  - Is a perfect cipher
  - Requires: changing key at each encryption
  - Key length = message length
  - Unfortunately, this key length is optimal

Part II
OTP with PRG

PRGs

- Pseudorandom generator

  PRG: $\{0,1\}^m \rightarrow \{0,1\}^n$,   for m ≪ n

- $s \overset{\$}{\leftarrow} \{0,1\}^m$

  $b \overset{\$}{\leftarrow} \{0,1\}$

  $d \leftarrow A^{Gen_b()}(m, n)$

  A wins iff $d = b$

```
Gen_b()
•  If $b = 1$, return $x \overset{\$}{\leftarrow} \{0,1\}^n$
•  Else, return  $x \leftarrow \mathrm{PRG}(s)$
```

- **<u>($\boldsymbol{k, \epsilon}$)-secure PRG</u>**:

  A pseudorandom generator PRG is (k, ϵ)-secure if, and only if, an adversary making at most k queries to Gen_b wins w.p. at most $\frac{1}{2} + \epsilon$

A relaxation of Perfection

- Security of perfect ciphers does not depend on the attacker's computational resources
  - Attacker with 200 years of computation time still learns nothing from ciphertext

- … however, we need very large keys

- We want smaller keys, but sufficient security
  - Idea: bound the adversary's resources
  - Allow some (small) information leakage
  - Adversary can "win" with very small proability

Less-than-perfect ciphers

- Now assume that we take $|\mathcal{K}| < |\mathcal{M}|$
- This introduces some attacks

- Meaningful message, random key:
  - Try to decrypt ciphertext with any possible key
  - This yields a list of “meaningful” possible plaintexts

- Compare to perfect security
  - PS: a ciphertext can hide any meaningful message
  - Imperfect security: ciphertext can “hide” at most $|\mathcal{K}|$ messages, with $|\mathcal{K}| < |\mathcal{M}|$
  - Key length determines security

Computational Security Basics

- Generic cipher family, depends on “sec. parameter” n
  - Usually the length of the secret key

- Encryption and Decryption are generic algorithms

- Cipher is secure if any adversary A can “break” the encryption scheme with negligible probability
  - Smaller than $^1/_{\mathrm{Poly}[n]}$ for any polynomial Poly$[n]$

Negligible probabilities

- What is negligible in theory?
  - Our favourite: 2^{-n}
  - Second best: $Poly[n] \cdot 2^{-n}$
  - Another possibility: $2^{-\log[n]}$ is non-negligible, but $2^{-\log^2[n]}$ is negligible

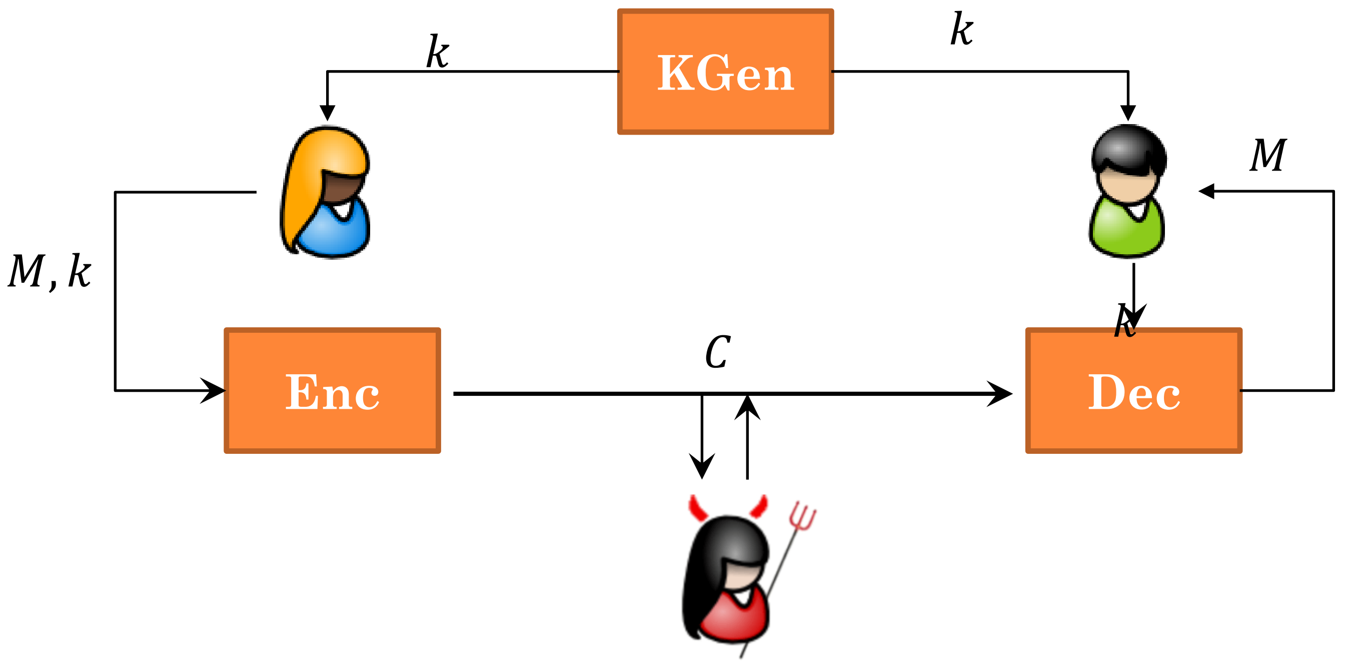
- What is negligible in practice?
  - Say the adversary wins with probability 2^{-n} for a small value of n
  - Trying again and again over a large amount of data, say 1GB, will eventually let $\mathcal{A}$ succeed
  - In practice, we like a security of at least 2^{-80}

Computational cipher security

- Think of it in terms of a game
- The adversary plays this game against our cipher and the parties using it – encryptor, decryptor
- A can see ciphertexts (polynomially many of them)
- Security notion: indistinguishability (of ciphertexts) from random

What is symmetric encryption

- Tuple of algorithms $(\mathrm{KGen}, \mathrm{Enc}, \mathrm{Dec})$ such that:
  - $\mathrm{KGen}(1^\gamma)$ outputs symmetric key k
  - $\mathrm{Enc}(\mathrm{k}, \mathrm{M})$ outputs cyphertext C
  - $\mathrm{Dec}(\mathrm{C}, \mathrm{k})$ outputs plaintext M



Semantic security (sym. encryption)

- Also called: IND-CPA – indistinguishability against chosen plaintext attacks

- Adversary plays against challenger
  - Challenger chooses key
  - Challenger chooses a bit b
  - Adversary can query $Enc(M)$ oracle, returns $\mathrm{Enc(k, M)}$
  - Test: A chooses messages m_0, m_1 such that $|m_0| = |m_1|$
  - Challenger returns $\mathrm{Enc(k, m_b)}$
  - A can go on querying Enc oracle
  - Finally, A outputs guess d of b

Exercise: try to write this def. in game form!

The IND-CPA game

- $k \overset{\$}{\leftarrow} \mathrm{KGen}(1^\gamma)$

  $b \overset{\$}{\leftarrow} \{0,1\}$

  $(\mathrm{m}_0, \mathrm{m}_1) \leftarrow A^{\mathrm{Enc}()}(\gamma)$ with $|m_0| = |m_1|$

  $c \leftarrow \mathrm{Enc}(k, m_b)$

  $d \leftarrow A^{\mathrm{Enc}()}(\gamma, c)$

  A wins iff $d = b$

- **(q, ϵ)-secure Symmetric Encryption**:

  A symmetric-key encryption scheme SEnc is (q, ϵ)-secure if, and only if, an adversary making at most q queries to Enc wins w.p. at most $\frac{1}{2} + \epsilon$

LoR versus RoR

- Previous version of game is Left-or-Right IND-CPA
- There are some more versions:
  - Real or Random
  - Ask then Guess
  - Etc.

- In TD you will discover some of these

- Essence of IND-CPA: the output of an encryption function gives no advantage to know plaintext
  - A type of pseudorandomness as well…

Advantage & Unpredictability

- In PRG game the adversary's winning probability should not be larger than ${}^{1}/_{2} + \varepsilon$
  - We call $\Pr[A \text{ wins}] - {}^{1}/_{2}$ the advantage of $\mathcal{A}$

- Unpredictability theorem:
  - If $G: \{0,1\}^n \to \{0,1\}^m$ with $m > n$ is a bounded-secure PRG, then for a randomly chosen $s \overset{\$}{\leftarrow} \{0,1\}^n$, no poly-runtime algorithm $\mathcal{P}$ given the first j bits of $G(s)$ can predict the $(j+1)$-th bit w.p. $\frac{1}{2} + \varepsilon$ for $\varepsilon \notin \mathrm{Negl}[n]$

Perfect to Imperfect cipher

- Why would we want that?
  - Well, it's more efficient, since $|\mathcal{K}| < |\mathcal{M}|$

- Recall the OTP
  - Traditional OTP for $\mathcal{K} = \mathcal{M} = \{0,1\}^m$
    - Choose random $k \overset{\$}{\leftarrow} \mathcal{K}$
    - Encrypt message m to : $c \coloneqq k \oplus m$
    - Decrypt ciphertext c as: $\hat{m} \coloneqq c \oplus k$
  - Unconditionally secure…
  - … But:
    - Key can only be used one time
    - Key is as long as message

Perfect to Imperfect OTP using PRG

- Recall the OTP
  - Traditional OTP for $\mathcal{K} = \mathcal{M} = \{0,1\}^m$
    - Choose random $k \overset{\$}{\leftarrow} \mathcal{K}$
    - Encrypt message m to : $c := k \oplus m$
    - Decrypt ciphertext c as: $\hat{m} := c \oplus k$

- Now replace random key generation by PRG:
  - OTP for $\mathcal{M} = \{0,1\}^m$ with $\mathcal{K} = \{0,1\}^n$ and $n < m$
  - Use a bounded-secure PRG $G: \{0,1\}^n \rightarrow \{0,1\}^m$
    - KeyGen: choose (once) $k \overset{\$}{\leftarrow} \mathcal{K}$
    - Encrypt message m as $c := G(k) \oplus m$
    - Decrypt message as: $\hat{m} := c \oplus G(k)$

Perfect/Imperfect ciphers

- Perfect ciphers:

$$\text{Prob}[\text{ptext} = M \mid \text{ctext} = C] = \text{Prob}[\text{ptext} = M]$$

  - Alternatively:

    For any messages $M_1 \neq M_2$ and any ciphertext C :

$$\text{Prob}[\text{Enc}(*, M_1) = C] = \text{Prob}[\text{Enc}(*, M_2) = C]$$

- Semantic security of imperfect ciphers:
  - For $k \overset{\$}{\leftarrow} K,\ b \overset{\$}{\leftarrow} \{0,1\}$, and for any two messages m_0, m_1, no polynomial-time adversary $\mathcal{A}$ given $\text{Enc}_k(m_b)$ can output $d = b$ with probability ${}^1/_2 +\ \varepsilon$ for $\varepsilon \notin \text{Negl}[|\mathcal{K}|]$

Our Imperfect OTP with PRG works!

- Theorem:
  - The OTP + PRG cipher we considered is q-semantically secure as long as the PRG is q-bounded-secure
  - Formally: for any adversary $\mathcal{A}$ against the q-semantic security of OTP+PRG, there exists a q-bounded adversary $\mathcal{B}$ against the PRG-security of G such that:

$$\Pr[\mathcal{A}\ wins] \leq \Pr[\mathcal{B}\ wins]$$

$$\boxed{\text{If OTP + PRG is insecure, then } G \text{ is insecure}}$$

$$\cong$$

$$\boxed{\text{As long as } G \text{ is secure, OTP + PRG is secure}}$$

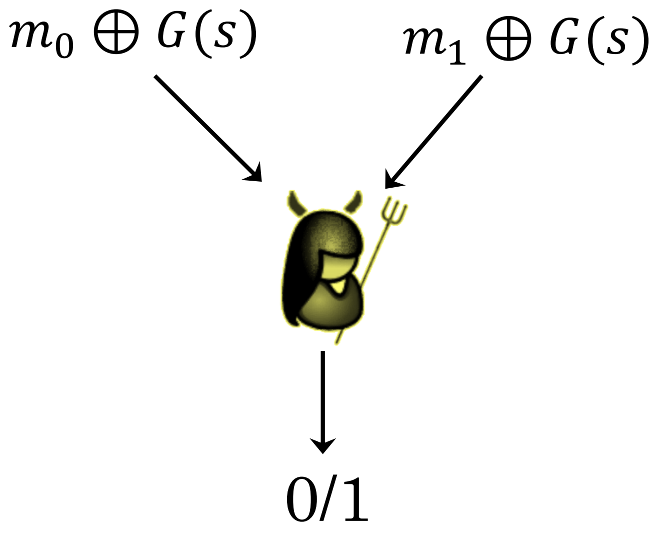
Proofs by game hopping

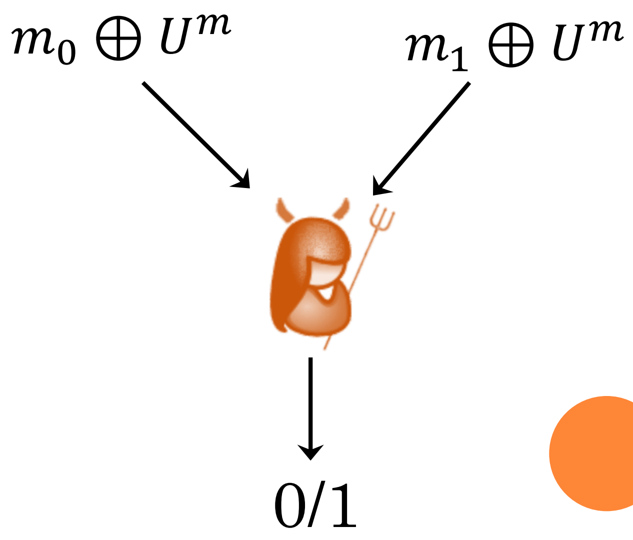
- Technique of game hopping (Shoup, 1999):
  - Start from original security game, G_0
  - Modify it to “restricted” game G_1
  - Argue that $G_0 \approx G_1$
  - Continue till last game can only be won by trivial A

- Game hops:
  - Restrict use of an oracle
  - Return different output to that of oracle
  - Restrict number of (honest) participants
  - …

LET'S PROVE THIS

- Proof:
  - Game 0: original semantic security game
  - Game 1: replace $G(s)$ by U^m in encryption
  - What kind of game equivalence do we need?





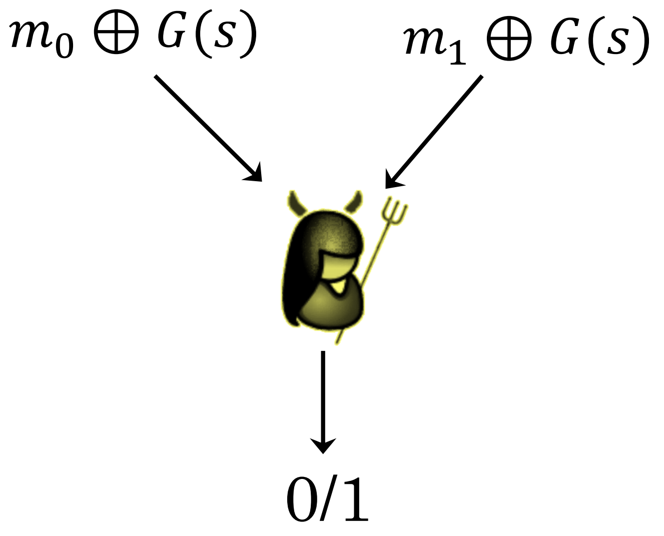
Game equivalence

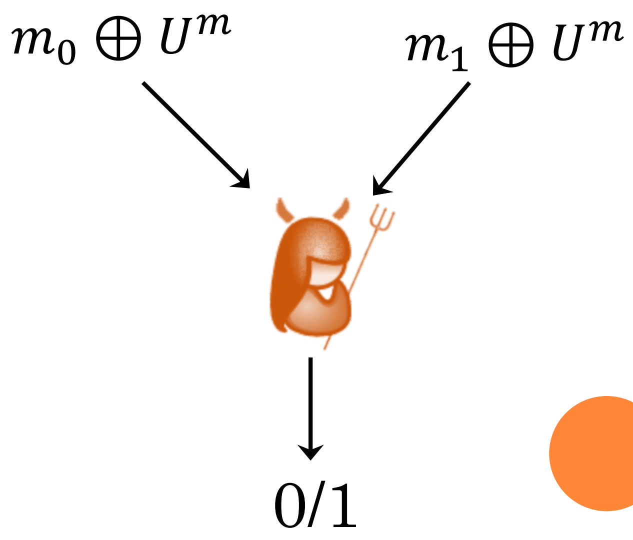
- Proving games equivalent:
  - A matter of deciding winning probability
  - Game hop should not alter A's winning probability
  - … in fact, alteration exists, but with negl. probability

- Some ways of proving equivalence:
  - Prove that A can make a specific query only by accident
    - So, with the exception of an accident, A does not make query
  - Prove that reduction can simulate well some queries
    - Which means, challenger must only handle the rest
  - Prove that the adversary cannot distinguish btw. outputs

Back to the proof

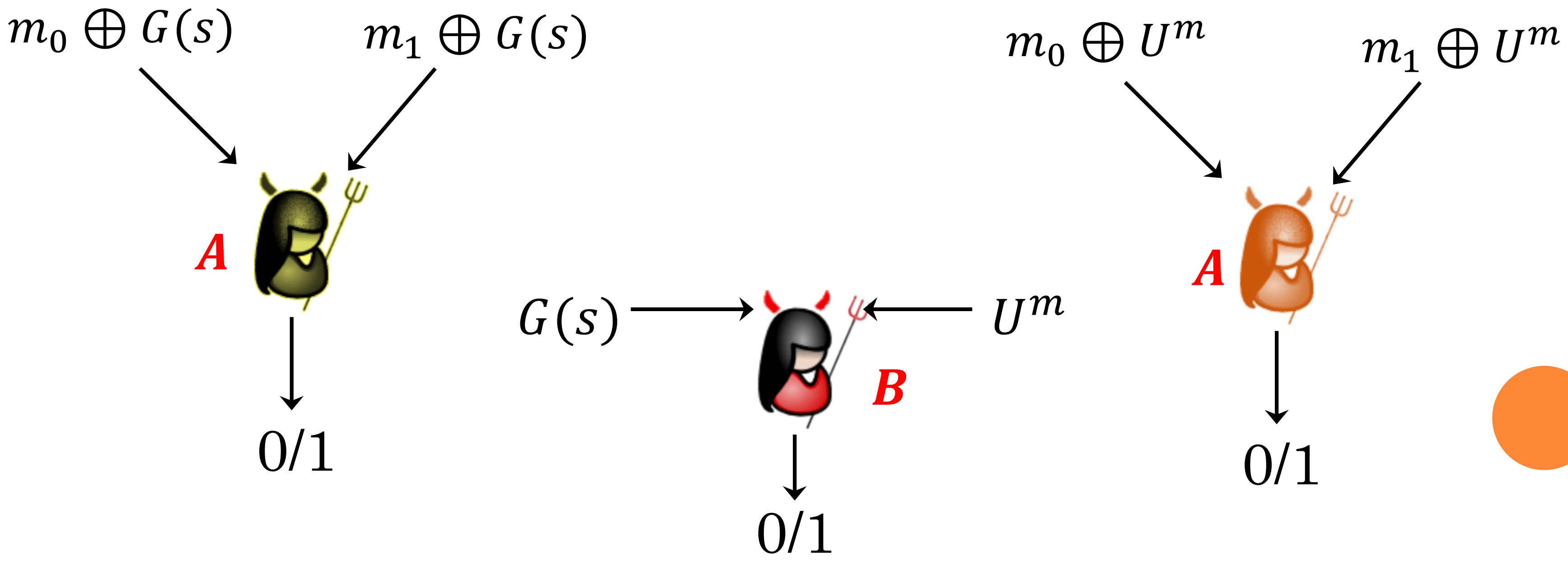
- Proving these games equivalent
  - A cannot distinguish between left and right output
  - Equivalent to proving: if $\mathcal{A}$'s game is altered, then there exists a distinguisher between $G(s)$ and U^m





LET'S PROVE THIS

- Proof:
  - Claim: if there exists a distinguisher $\mathcal{D}$ between games, then we can construct PRG adversary $\mathcal{B}$ from $\mathcal{D}$
  - What is a distinguisher?



A DISTINGUISHING GAME

- In our case, G_0, G_1 are equivalent except output
- Distinguisher plays against a challenger
  - Its goal is to distinguish G_0 from G_1, not to win them
  - Challenger first sets up games (choose seed, bit b)
  - Then challenger also picks additional bit b*
  - Adversary plays normally
  - For Encrypt queries: Chg returns output from G_{b^*}
  - Finally A will need to output guess d^* of b^*

- Constructing the reduction:
  - Show that if D can distinguish G_0 from G_1, then we can construct B that distinguishes $G(s)$ from U^m

Who plays what game

- D plays distinguishing game:
  - $k \overset{\$}{\leftarrow} \mathrm{KGen}(1^{\gamma})$

    $b, b^* \overset{\$}{\leftarrow} \{0,1\}$

    $(\mathrm{m}_0, \mathrm{m}_1) \leftarrow A^{\mathrm{Enc}()}(\gamma)$ with $|m_0| = |m_1|$

    $c \leftarrow \mathrm{Enc}(k, m_b)$

    $d^* \leftarrow A^{\mathrm{Enc}()}(\gamma, c)$

    A wins iff $d^* = b^*$

- B plays PRG game:
  - $s \overset{\$}{\leftarrow} \{0,1\}^m$

    $b \overset{\$}{\leftarrow} \{0,1\}$

    $d \leftarrow A^{Gen_b()}(m, n)$

    A wins iff $d = b$

$\boxed{\begin{array}{l} \mathrm{Gen}_b() \\ \bullet\ \text{If } b = 1\text{, return } x \overset{\$}{\leftarrow} \{0,1\}^n \\ \bullet\ \text{Else, return } x \leftarrow \mathrm{PRG}(s) \end{array}}$

Constructing the reduction B

$$\begin{tikzcd}
C_B & B = C_D & D \\
\begin{array}{l} s \overset{\$}{\leftarrow} \{0,1\}^m \\ b^* \overset{\$}{\leftarrow} \{0,1\} \end{array} \arrow[r, "\text{Setup done}"] & b \overset{\$}{\leftarrow} \{0,1\} \arrow[r, "\text{Setup done}"] & {} \\
{} & {} \arrow[l, "\text{Query Gen}_b"'] & {} \arrow[l, "\text{Enc(m)}"'] \\
\begin{array}{l} \bullet\ \text{If } b^* = 1, \text{ return } x \overset{\$}{\leftarrow} \{0,1\}^n \\ \bullet\ \text{Else, return } x \leftarrow \mathrm{G}_{\text{small}}(s) \end{array} \arrow[r, "x"] & C = m \oplus x \arrow[r, "\text{C}"] & {} \\
{} & {} \arrow[l, "\text{Query Gen}_b"'] & {} \arrow[l, "m_0, m_1"'] \\
{} \arrow[r, "x^*"] & C^* = m_b \oplus x^* \arrow[r, "C^*"] & {} \\
{} & {} \arrow[l, "\text{Output } d^*"'] & {} \arrow[l, "\text{Guess bit } d^*"']
\end{tikzcd}$$

ANALYSIS OF REDUCTION

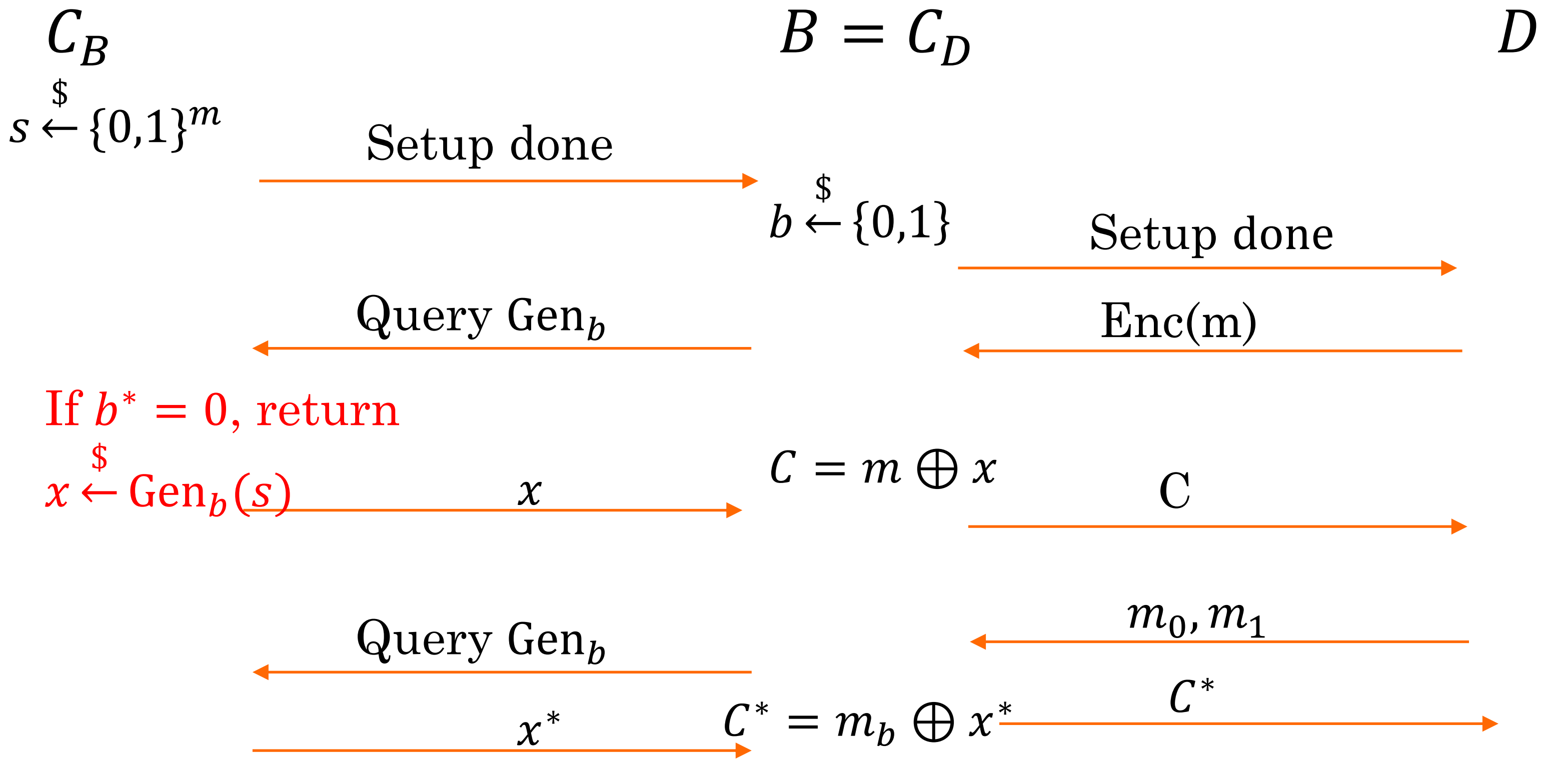
- If C_B drew the bit $\mathrm{b}^* = 1$ then D sees:

$$\begin{tikzcd}
C_B & B = C_D & D \\
s \overset{\$}{\leftarrow} \{0,1\}^m \arrow[r, "\text{Setup done}"] & b \overset{\$}{\leftarrow} \{0,1\} \arrow[r, "\text{Setup done}"] & {} \\
{} & {} \arrow[l, "\text{Query Gen}_b"'] & {} \arrow[l, "\text{Enc(m)}"'] \\
\bullet \text{ If } b^* = 1 \text{, return } x \overset{\$}{\leftarrow} \{0,1\}^n \arrow[r, "x"] & C = m \oplus x \arrow[r, "\text{C}"] & {} \\
{} & {} \arrow[l, "\text{Query Gen}_b"'] & {} \arrow[l, "m_0, m_1"'] \\
{} \arrow[r, "x^*"] & C^* = m_b \oplus x^* \arrow[r, "C^*"] & {}
\end{tikzcd}$$

This is the output of game G_1

Analysis of reduction

- If C_B drew the bit $b^* = 0$ then D sees:



This is the output of game G_0

Concluding anaylsis

- Say that D wins w.p. $\frac{1}{2} + \epsilon_D$
- Then B wins with the same probability
- Conclusion for proof:

$$\begin{aligned}\Pr[A \text{ wins } G_0] &\leq \Pr[A \text{ wins } G_1] + \epsilon_D \\ &= \Pr[A \text{ wins } G_1] + \mathrm{Adv}_{\mathrm{PRG}}[B]\end{aligned}$$

- Winning G_1:
  - A has to distinguish between $m_0 \oplus r_0$ and $m_1 \oplus r_1$
  - But this time r_0, r_1 truly random
  - Winning this game is equivalent to distinguishing between two truly random numbers
  - Thus, $\Pr[A \text{ wins } G_1] = \frac{1}{2}$

Conclusion of proof

- Proof:
  - Game 0: original semantic security game
  - Game 1: replace $G(s)$ by U^m in encryption
  - Winning game 1: probability of ½

$$\begin{aligned}\Pr[A \text{ wins } G_0] &\leq \Pr[A \text{ wins } G_1] + \epsilon_D \\ &= \Pr[A \text{ wins } G_1] + (\Pr[D \text{ dist. } G_0 \text{ from } G_1\,] - \tfrac{1}{2}) \\ &= \frac{1}{2} + \left(\Pr[B \text{ wins}] - \frac{1}{2}\right) = \Pr[B \text{ wins}].\end{aligned}$$

IND-CPA ENCRYPTION IN PERSPECTIVE

- PRG + OTP construction is IND-CPA secure

- IND-CPA-secure encryption is also PRF-secure
- Pseudorandom functions:

$$\mathrm{F}: Keys \times Input \rightarrow \{0,1\}^*$$

- Security:
  - Bounding output length to n bits

$$K \overset{\$}{\leftarrow} Keys$$
$$b \overset{\$}{\leftarrow} \{0,1\}$$
$$\underline{d \leftarrow A^{Gen_b(x)}(Keys, Input)}$$
$$A \text{ wins iff } d = b$$

$\boxed{\begin{array}{l} \mathrm{Gen}_b(x) \\ \bullet\ \text{If } b = 1, \text{ return } x \overset{\$}{\leftarrow} \{0,1\}^n \\ \bullet\ \text{Else, return } x \leftarrow \mathrm{F}_K(x) \end{array}}$

Conclusion for today

Models and proofs

- More security models:
  - IND-CPA security (left-or-right version)
  - Pseudorandom function (PRF)
- Construction:
  - IND-CPA/PRF from PRG (and OTP)
- Proofs:
  - Game hop technique
  - Distinguishing between games

Questions?

Pseudorandomness

- What is a "random" string?
  - Usually defined as a string for which the probability that any of the bits is 1 is exactly ½
- How does the attacker distinguish in practice?
  - Fixed bits
  - Fixed relationship between bits
  - Un-fixed, but biased relationship between bits (occurring with prob. p, such that $|p - {}^{1}/_{2}|$ non-negligible)
  - **Theorem:** In a random string, the probability that there are less than ${}^{|m|}/_{3}$ bits equal to 1 is negligible
    - Proof in TD

Statistical Tests

- Theorem:
  - Consider $\mathcal{T}_{m,k}$ to be the poly-sized set of all statistical tests $T_{m,k}$ which have poly-runtime, which take as input a sample of k bitstrings of length m, for a known, fixed $k \in \text{Poly}[m]$ and which output 0 (if the string sample is not random) and 1 (if the string sample is random)
  - Assume that we have a PRG $G: \{0,1\}^n \to \{0,1\}^m$ for $m = 2n$
  - Then: G is a secure PRG against a $k-$bounded adversary $\mathcal{A}$ if, and only if, for all $T_{m,k} \in \mathcal{T}_{m,k}$ it holds that for $s \overset{\$}{\leftarrow} \{0,1\}^n$, $T_{m,k}$ run on randomly chosen k-sized samples of $G(s)$ returns 0 w.p. at most $\varepsilon \in \text{Negl}[m]$

Proof by reduction

- Theorem:
  - Assume $T_{m,k} \in \mathcal{T}_{m,k}$ with input a sample of k bitstrings of length m, outputting 0 (if not random) and 1 (if random)
  - Assume $G: \{0,1\}^n \to \{0,1\}^m$ for $m = 2n$
  - Then: G is $k-$bounded secure iff. for $s \overset{\$}{\leftarrow} \{0,1\}^n$, $\forall\, T_{m,k} \in \mathcal{T}_{m,k}$ run on the output dist. of G returns 0 w.p. $\varepsilon \in \text{Negl}[m]$

- Proof : $\Rightarrow$
  - Say G is k-bounded secure PRG
  - Assume $\exists\, T_{m,k} \in \mathcal{T}_{m,k}$ which returns 0 w.p. $\delta \notin \text{Negl}[m]$
  - Claim: $\delta \notin \text{Negl}[n]$. Why is this true?
  - Construct k-bounded $\mathcal{A}$ against k-bounded sec. of G s.t. $\mathcal{A}$ wins with probability $p_{\mathcal{A}} \notin \text{Negl}[n]$

Proof by reduction

- Theorem:
  - Assume $T_{m,k} \in \mathcal{T}_{m,k}$ with input a sample of k bitstrings of length m, outputting 0 (if not random) and 1 (if random)
  - Assume $G: \{0,1\}^n \rightarrow \{0,1\}^m$ for $m = 2n$
  - Then: G is $k-$bounded secure iff. for $s \overset{\$}{\leftarrow} \{0,1\}^n$, $\forall\, T_{m,k} \in \mathcal{T}_{m,k}$ run on the output dist. of G returns 0 w.p. $\varepsilon \in \mathrm{Negl}[m]$
- Proof : $\Rightarrow$
  - $\mathcal{A}$ plays the PRG game. First the game picks: $s \overset{\$}{\leftarrow} \{0,1\}^n$ bit b
  - Query Gen_b k times (ok, $\mathcal{A}$ is k-bounded), get $\mathrm{X} = \{x_1, \dots, x_k\}$
  - Run $T_{m,k}$ on X, get output $d \in \{0,1\}$ (ok, test has poly-runtime)
    - If $\mathcal{A}$ does not know which test is good, it can run all of them
  - Return output d to PRF game
    - If $\mathcal{A}$ tried all tests, return min of all d values

Proof by reduction

- Proof :
  - $\mathcal{A}$ plays the PRG game. First the game picks: $s \overset{\$}{\leftarrow} \{0,1\}^n$ bit b
  - Query $Gen_b()$ k times (ok, $\mathcal{A}$ is k-bounded), get $X = \{x_1, \dots, x_k\}$
  - Run $T_{m,k}$ on X, get output $d \in \{0,1\}$ (ok, test has poly-runtime)
  - Return output d to PRF game
- Analysis:
  - Obs 1: $T_{m,k}$ always returns 1 if bit $b = 1$ ($x_1, \dots, x_k$ random)
  - Obs 2: if $b = 0$ then X contains outputs of G. Then $T_{m,k}$ returns 0 w.p. $\delta \notin \text{Negl}[m]$ (by assumption)
  - A wins w.p. $\Pr[A \text{ wins} \mid b = 1] \cdot \Pr[b = 1] + \Pr[A \text{ wins} \mid b = 0] \cdot \Pr[b = 0] = {}^1/_2 + {}^1/_2\ \delta,$ with $\delta \notin \text{Negl}[n]$
  - So G not a secure PRG. Contradiction

Food for thought

- Some significant proof steps:
  - $\text{Negl}[m] \cong \text{Negl}[n]$
    - Requiring $m \in \text{Poly}[n]$
  - $\mathcal{T}_{m,k}$ requires a sample of k elements
    - Requiring that our $\mathcal{A}$ is at least k-bounded!
  - $\mathcal{T}_{m,k}$ runs in polynomial time
    - Else, a bounded adversary cannot run this test
  - Statement about test holds for randomly chosen seed
    - If it held only for some seeds, we would not be able to transfer winning probability (PRG game first picks seed at rnd.)
    - We could have said it held for ALL keys. But then, it would not be an iff. statement. Let's see why.

Now the other way

- Theorem:
  - Assume $T_{m,k} \in \mathcal{T}_{m,k}$ with input a sample of k bitstrings of length m, outputting 0 (if not random) and 1 (if random)
  - Assume $G: \{0,1\}^n \to \{0,1\}^m$ for $m = 2n$
  - Then: G is k−bounded secure iff. for $s \overset{\$}{\leftarrow} \{0,1\}^n$, $\forall\, T_{m,k} \in \mathcal{T}_{m,k}$ run on the output dist. of G returns 0 w.p. $\varepsilon \in \mathrm{Negl}[m]$

- Proof : $\Leftarrow$
  - Say $\forall\, T_{m,k} \in \mathcal{T}_{m,k}$ returns 0 w.p. at most $\delta \in \mathrm{Negl}[m]$
  - Say $\exists$ k−bounded A winning w.p. $1/2 + \varepsilon \notin \mathrm{Negl}[n]$
  - Again $\varepsilon \notin \mathrm{Negl}[m]$
  - Construct poly-time test $T_{m,k}$ that outputs 0 w.p. $p_T \notin \mathrm{Negl}[m]$
    - Claim: $\mathcal{A}$ is that $T_{m,k}$

Pseudorandomness

- Intuition:
  - If A can’t tell ciphertexts from completely random strings of the same lengths, then:
    - A can’t see a plaintext/ciphertext dependence
    - A can’t see a key/ciphertext dependence

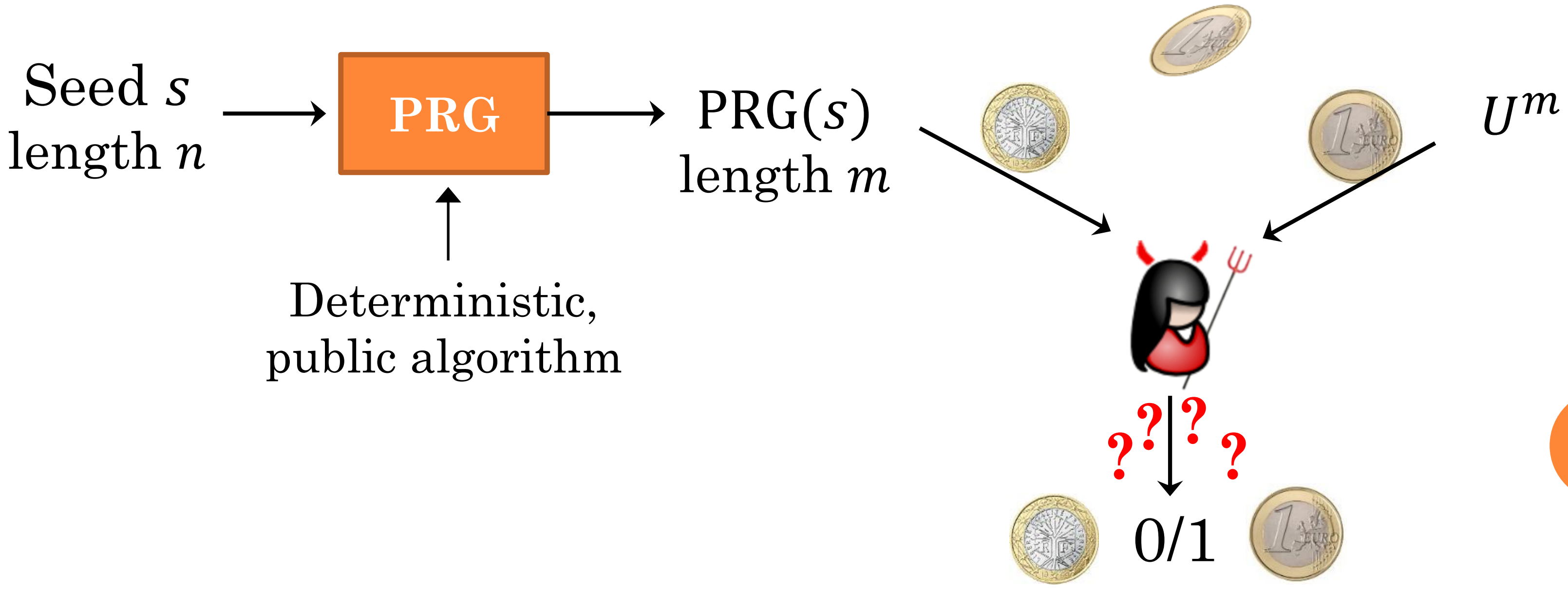
- Indistinguishability of real cryptographic systems from their idealizations is fundamental to provable security

Pseudorandom Generators (PRG)

- Principle: start from a small, random string (called a seed), get a larger string that looks random

$$\text{PRG} : \{0,1\}^n \to \{0,1\}^m \quad \text{for } m > n$$

- Security: a "good" PRG outputs strings that are indistinguishable from random (by an adversary)



The Secure-PRG game

- $s \overset{\$}{\leftarrow} \{0,1\}^n$

  $b \overset{\$}{\leftarrow} \{0,1\}$

  $d \leftarrow \mathcal{A}^{Gen_b()}(m, n, \mathrm{PRG})$

  $\mathcal{A}$ wins iff. $b = d$

```
Gen_b():
If $b = 1$ then $x \overset{\$}{\leftarrow} U^m$
Else $x \leftarrow \mathrm{PRG}(s)$
Return $x$
```

- Unbounded vs. bounded $\mathcal{A}$
  - Unbounded: as many calls to Gen_b as $\mathcal{A}$ wants
  - Bounded: only polynomially many calls, poly-runtime
    - k-bounded: only k calls, poly-runtime

- **(k, ε)-Secure PRG**: G is a k-bounded-secure PRG if, and only if, any k-bounded adversary $\mathcal{A}$ wins w.p. at most $^1/_2 + \varepsilon$
  - (asymptotically) k-secure: $\varepsilon \in \mathrm{Negl}[n]$