The Fundamental Lemma of Game Playing

Advanced Topics in ~~Cybersecurity~~Cryptography (7CCSMATC)

Martin R. Albrecht

Outline

Introduction

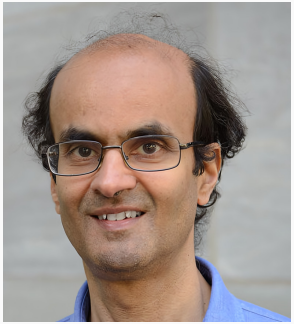
Recap

- We have defined what it means for an encryption scheme to be secure (IND-CPA + INT-CTXT = IND-CCA).
- We have shown that the OTP achieves IND-CPA security, even unconditionally.

The One-Time Pad is impractical, we want something more manageable $\Rightarrow$ Pseudorandomness!

Main Reference

Mihir Bellare and Phillip Rogaway. **Code-Based Game-Playing Proofs and the Security of Triple Encryption.** Cryptology ePrint Archive, Report 2004/331. 2004. URL: `https://eprint.iacr.org/2004/331`

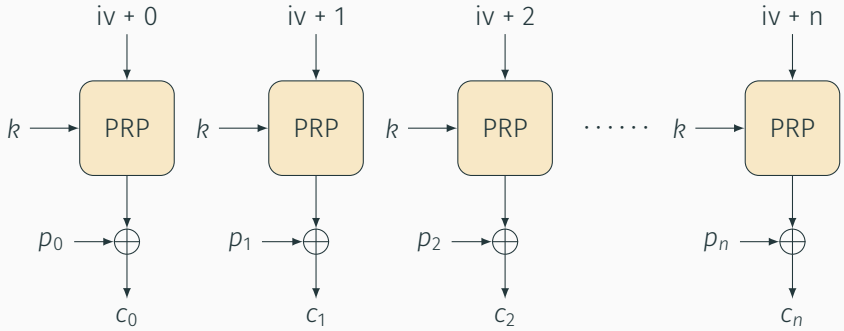


Mihir Bellare is a professor at UCSD

- 2003 RSA Conference's Sixth Annual Award
- 2013 Fellow of the Association for Computing Machinery.
- 2019 Levchin Prize for Real-World Cryptography

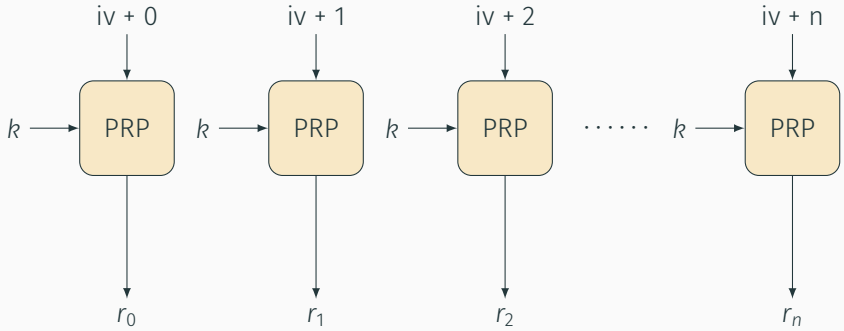
CTR Mode

CTR Mode



Picture credit: https://www.iacr.org/authors/tikz/

CTR Mode Stream



$r_i \in \{0,1\}^\lambda$

Want: $n+1$ Pseudorandom Strings of length λ

Definition (PRF)

A PRF is a keyed function $F_k : \{0,1\}^\lambda \to \{0,1\}^N$ where N depends on λ and for $k \leftarrow\!\$\ \mathcal{K}$. We say F_k is (t, ε)-secure **PRF** if for Game_0 and Game_1 defined below we have:

$$\forall\, \mathcal{D} \in t \text{ steps: } \mathsf{Adv}_F^{\mathrm{prf}}(\mathcal{D}) = \left|\Pr[\mathcal{D}^{\mathsf{Game}_1} = 1] - \Pr[\mathcal{D}^{\mathsf{Game}_0} = 1]\right| < \varepsilon$$

Game_0

1 : $f \leftarrow \emptyset$

2 : **return** $\mathcal{D}^{\mathsf{F}}$

Game_1

1 : $f \leftarrow \emptyset$; $k \leftarrow\!\$\ \mathcal{K}$

2 : **return** $\mathcal{D}^{\mathsf{F}}$

$\mathsf{F}(x)$

1 : **if** $x \notin f.\mathrm{keys}$ **then** $f[x] \leftarrow\!\$\ \{0,1\}^N$

2 : $y \leftarrow f[x]$

3 : $y \leftarrow F_k(x)$ //Game_1

4 : **return** y

Have: $n+1$ calls to Pseudorandom Permutation of length λ

Definition (PRP)

A PRP is a keyed permutation $E_k : \{0,1\}^\lambda \to \{0,1\}^\lambda$ for $k \leftarrow\!\$\, \mathcal{K}$. We say E is (t, ε)-secure **PRP** if for Game_0 and Game_1 defined below we have:

$$\forall\, \mathcal{D} \in t \text{ steps: } \text{Adv}^{\text{prp}}_E(\mathcal{D}) = \left|\Pr[\mathcal{D}^{\text{Game}_1} = 1] - \Pr[\mathcal{D}^{\text{Game}_0} = 1]\right| < \varepsilon$$

Game_0

1 : $f \leftarrow \emptyset$

2 : **return** $\mathcal{D}^{\text{P}}$

Game_1

1 : $f \leftarrow \emptyset$; $k \leftarrow\!\$\, \mathcal{K}$

2 : **return** $\mathcal{D}^{\text{P}}$

$\text{P}(x)$

1 : **if** $x \notin f.\text{keys}$ **then** $f[x] \leftarrow\!\$\, \{0,1\}^\lambda \setminus f.\text{values}$

2 : $y \leftarrow f[x]$

3 : $y \leftarrow E_k(x)$ //Game_1

4 : **return** y

The Gap

Game_0

1 : $f \leftarrow \emptyset$

2 : **return** $\mathcal{D}^{\mathsf{F}}$

Game_1

1 : $f \leftarrow \emptyset$;

2 : **return** $\mathcal{D}^{\mathsf{F}}$

$\mathsf{F}(x)$

1 : **if** $x \in f.\mathrm{keys}$ **then**

2 : $\quad y \leftarrow f[x]$

3 : **else**

4 : $\quad y \leftarrow\!\$\ \{0,1\}^{\lambda} \setminus f.\mathrm{values}$

5 : $\quad y \leftarrow\!\$\ \{0,1\}^{\lambda}$ //Game_1

6 : $f[x] \leftarrow y$

7 : **return** y

PRP-PRF Switching Lemma

Lemma

Let π be a random ***permutation*** *from $\{0,1\}^\lambda \to \{0,1\}^\lambda$; let ρ be a random* ***function*** *from $\{0,1\}^\lambda \to \{0,1\}^\lambda$. Let $\mathcal{A}$ be an adversary making at most q queries to its oracle, then:*

$$|\Pr[\mathcal{A}^\pi] - \Pr[\mathcal{A}^\rho]| \leq \frac{q \cdot (q-1)}{2^{\lambda+1}}.$$

PRP-PRF Switching Lemma I

Consider the following games:

Game_0

1 : $\pi \leftarrow \emptyset$
2 : **return** $\mathcal{A}^{\mathrm{P}}$

Game_1

1 : $\pi \leftarrow \emptyset$
2 : **return** $\mathcal{A}^{\mathrm{P}}$

$\mathrm{P}(x)$

1 : $y \leftarrow\!\$\ \{0,1\}^\lambda$
2 : **if** $y \in \pi.\mathrm{values}$ **then**
3 : $\quad \mathrm{bad} \leftarrow \mathbf{true}$
4 : $\quad y \leftarrow\!\$\ \{0,1\}^\lambda \setminus \pi.\mathrm{values}$ // Game_1
5 : $\pi[x] \leftarrow y$
6 : **return** y

PRP-PRF Switching Lemma II

$$
\begin{aligned}
|\Pr[\mathcal{A}^{\pi}] - \Pr[\mathcal{A}^{\rho}]| &= \left|\Pr[\mathcal{A}^{\mathsf{Game}_0}] - \Pr[\mathcal{A}^{\mathsf{Game}_1}]\right| && (1)\\
&\leq \Pr[\mathcal{A}^{\mathsf{Game}_0}] \text{ sets bad} && (2)\\
&\leq q \cdot (q+1)/2^{\lambda+1} && (3)
\end{aligned}
$$

On Eq. (1): Game_0 perfectly simulates a random function ρ and Game_1 perfectly simulates a random permutation π, by the principle of lazy sampling. Thus, we have

$$\Pr[\mathcal{A}^{\rho}] = \Pr[\mathcal{A}^{\mathsf{Game}_1}] \text{ and } \Pr[\mathcal{A}^{\mathsf{Game}_2}] = \Pr[\mathcal{A}^{\pi}].$$

On Eq. (2): we will appeal to the fundamental lemma of game playing.

On Eq. (3): by the union bound the probability that $y \in \pi.\mathsf{values}$, is at most

$$\frac{(1+2+\cdots+(q-1))}{2^{\lambda}} = \frac{q\cdot(q-1)}{2^{\lambda+1}}.$$

Game Playing

We say Game_0 and Game_1 are "identical-until-bad" if they are … identical until some flag bad is set.

Termination and Finite Randomness I

- We assume that an underlying execution model provides a notion for the number of steps (the running time) of a program.
- We require that both the adversary and the game always terminate in finite time.
  - For any adversary $\mathcal{A}$ there must exist an integer T such that $\mathcal{A}$ always halts within T steps (regardless of the random choices $\mathcal{A}$ makes and the answers it receives to its oracle queries).
  - For any game Game there must exist an integer T such that Game always halts within T steps (regardless of the random choices made).

Termination and Finite Randomness II

- Since $\mathcal{A}$ and Game terminate in finite time,
  - there must be an integer T such that they each execute at most T random-assignment statements, and
  - there must be an integer B such that the size of the set $\mathcal{S}$ in any random-assignment statement $s \leftarrow\!\!\$\ \mathcal{S}$ executed by the adversary or the game is at most B.

$\Rightarrow$ The execution of Game with $\mathcal{A}$ uses finite randomness, meaning Game and $\mathcal{A}$ are underlain by a finite sample space Ω.

Punchline

Probabilities are well-defined and we can talk about the probabilities of various events in the execution.

Fundamental Lemma of Game Playing

Fundamental Lemma of Game Playing

Lemma (Fundamental Lemma of Game Playing)

Let Game_0, Game_1, Game_2 *be identical-until-bad games and* $\mathcal{A}$ *be an adversary. Then*

$$\left|\Pr[\mathcal{A}^{\mathsf{Game}_0}] - \Pr[\mathcal{A}^{\mathsf{Game}_1}]\right| \leq \Pr[\mathcal{A}^{\mathsf{Game}_2} \textit{ sets bad}] \textit{ and}$$

$$\left|\Pr[\mathsf{Game}_0{}^{\mathcal{A}}] - \Pr[\mathsf{Game}_1{}^{\mathcal{A}}]\right| \leq \Pr[\mathsf{Game}_2{}^{\mathcal{A}} \textit{ sets bad}].$$

- The first statement follows immediately from the second.
- For the second statement we first prove it with $\mathsf{Game}_2 = \mathsf{Game}_0$ and then generalise.

Proof I

- We have required that the adversary and game always terminate in finite time, and also that there is an integer that bounds the size of any set $\mathcal{S}$ in any random-assignment statement $s \xleftarrow{\$} \mathcal{S}$ executed by the adversary or game.
- This means that there exists an integer z such that the execution of Game_0 with $\mathcal{A}$ and the execution of Game_1 with $\mathcal{A}$ perform no more than z random-assignment statements, each of these sampling from a set of size at most z.

Proof II

- Let $\mathcal{C} := \mathsf{Coins}(\mathcal{A}, \mathsf{Game}_0, \mathsf{Game}_1) = [1 \dots z!]^z$ be the set of z-tuples of numbers, each number between 0 and $z!$.

```
z = 2
R = IntegerModRing(factorial(z)); offset = vector(R, z, [1]*z).lift()
Coins = [coin.lift() + offset for coin in FreeModule(R, z)]
print(Coins)
```

```
[(1, 1), (2, 1), (1, 2), (2, 2)]
```

- For $\mathbf{c} = (c_0, \dots, c_{z-1}) \in \mathcal{C}$, the execution of Game with $\mathcal{A}$ on coins c is defined as follows:
  - On the i-th random-assignment statement, call it $x \leftarrow\!\$\, \mathcal{U}(\mathcal{S})$, where $\mathcal{S} := \{s_i\}_{0 \le i < m}$, if $\mathcal{S} \neq \emptyset$, return $s_{c_i \bmod |\mathcal{S}|}$, otherwise return $\perp$.
- This way to perform random-assignment statements is done regardless of whether it is $\mathcal{A}$ or one of the procedures from Game that is is performing the random-assignment statement.

Proof III

- Note that $m = |\mathcal{S}|$ satisfies $m|z!$ so if $\mathbf{c}$ is chosen at random from $\mathcal{C}$ then the mechanism above will return a point x drawn uniformly from $\mathcal{S}$, and also the values for each random-assignment statement are independent.

Proof iv

- For $\mathbf{c} \in \mathcal{C}$ we let $\mathrm{Game}_0{}^{\mathcal{A}}(\mathbf{c})$ denote the output of Game_0 when Game_0 is executed with $\mathcal{A}$ on coins $\mathbf{c}$. Same for Game_1.
- Write $\mathcal{C}_{i,\mathrm{one}} := \{\mathbf{c} \in \mathcal{C} : \mathrm{Game}_i{}^{\mathcal{A}}(\mathbf{c}) \Rightarrow 1\}$
- Write $\mathcal{C}_i^{bad} \subseteq \mathcal{C}$ for the coins that result in *bad* being set to **true** when running $\mathrm{Game}_i{}^{\mathcal{A}}$.
- Partition $\mathcal{C}_{i,\mathrm{one}}$ into $\mathcal{C}_{i,\mathrm{one}}^{bad}$ and $\mathcal{C}_{i,\mathrm{one}}^{good}$ depending on whether bad was set or not in game Game_i.
- Because games Game_0 and Game_1 are identical-until-bad, an element $\mathbf{c} \in \mathcal{C}$ is in $\mathcal{C}_{0,\mathrm{one}}^{good}$ if and only if it is in $\mathcal{C}_{1,\mathrm{one}}^{good}$.
- Bad is never set so the sets are same and in particular have the same size.

Proof v

We then get:

$$\begin{aligned}
\Pr[\mathsf{Game}_0{}^{\mathcal{A}}] - \Pr[\mathsf{Game}_1{}^{\mathcal{A}}] &= \frac{\mathcal{C}_{0,\mathrm{one}}}{\mathcal{C}} - \frac{\mathcal{C}_{1,\mathrm{one}}}{\mathcal{C}} \\
&= \frac{\mathcal{C}^{good}_{0,\mathrm{one}} + \mathcal{C}^{bad}_{0,\mathrm{one}}}{\mathcal{C}} - \frac{\mathcal{C}^{good}_{1,\mathrm{one}} + \mathcal{C}^{bad}_{1,\mathrm{one}}}{\mathcal{C}} \\
&= \frac{\mathcal{C}^{bad}_{0,\mathrm{one}}}{\mathcal{C}} - \frac{\mathcal{C}^{bad}_{1,\mathrm{one}}}{\mathcal{C}} \\
&\leq \frac{\mathcal{C}^{bad}_{0,\mathrm{one}}}{\mathcal{C}} \\
&\leq \frac{\mathcal{C}^{bad}_{0}}{\mathcal{C}} \\
&= \Pr[\mathsf{Game}_0{}^{\mathcal{A}} \text{ sets bad}].
\end{aligned}$$

Proof vi

To prove the second statement we rely on the following lemma.

Lemma

Let Game_0 *and* Game_1 *be identical-until-bad games. Let* $\mathcal{A}$ *be an adversary. Then*

$$\Pr[\mathsf{Game}_0{}^{\mathcal{A}} \text{ sets } bad] = \Pr[\mathsf{Game}_1{}^{\mathcal{A}} \text{ sets } bad].$$

Proof vii

- Since Game_0 and Game_1 are identical-until-bad, each $\mathbf{c} \in \mathcal{C}$ causes bad to be set in ${\mathsf{Game}_0}^{\mathcal{A}}$ iff it is set in ${\mathsf{Game}_1}^{\mathcal{A}}$.
- Thus

$$\begin{aligned}
\mathcal{C}_1^{bad} &= \mathcal{C}_2^{bad} \\
|\mathcal{C}_1^{bad}| &= |\mathcal{C}_2^{bad}| \\
|\mathcal{C}_1^{bad}|/|\mathcal{C}| &= |\mathcal{C}_2^{bad}|/|\mathcal{C}| \\
\mathsf{Pr}[{\mathsf{Game}_1}^{\mathcal{A}} \text{ sets bad}] &= \mathsf{Pr}[{\mathsf{Game}_2}^{\mathcal{A}} \text{ sets bad}].
\end{aligned}$$

Matching Attack

- Call $\sqrt{2^\lambda} = 2^{\lambda/2}$ times and check if any answer repeats.
- By the birthday bound this happens with constant probability

Memory-less Attack

Read about the Pollard-rho attack to learn how to make this attack use $\text{poly}(\lambda)$ memory instead of $2^{\lambda/2}$.

Fin

We want to approximate the one-time pad

If we have a PRF, this is straight-forward

If we "only" have a PRP, an ideal primitive, this breaks down after $q = \sqrt{2^\lambda}$ queries, e.g. 2^{64} for $\lambda = 128$ (AES-128).

Next: How do we get a PRP?

References I

[BR04] Mihir Bellare and Phillip Rogaway. Code-Based Game-Playing Proofs and the Security of Triple Encryption. Cryptology ePrint Archive, Report 2004/331. 2004. URL: https://eprint.iacr.org/2004/331.