Cryptography
CS 555

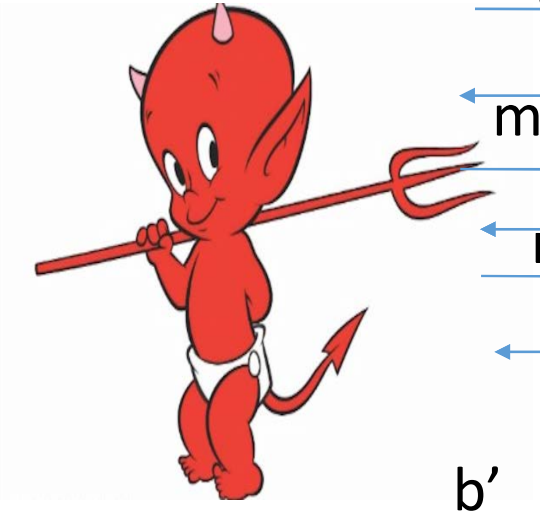
Topic 7: Pseudorandom Functions and CPA-Security

Recap

- Pseudorandom Generators G(s)
- Chosen Plaintext Attacks/CPA-Security
- ~~Build CPA-secure encryption scheme~~

- **Today's Goal:** Construct encryption scheme with CPA-security
- **Recall**: CPA-Security for single encryptions implies CPA-Security for multiple encryptions.

CPA-Security (Single Message)



m_0, m_1

$c = Enc_K(m_b)$

m_2

$c_2 = Enc_K(m_2)$

m_3

$c_3 = Enc_K(m_3)$

⋮

b'

Random bit b

K = Gen(.)

$$\forall PPT\ A\ \exists \mu\ (\text{negligible})\ \text{s.t}$$

$$\Pr[A\ \ Guesses\ b' = b] \le \frac{1}{2} + \mu(n)$$

Pseudorandom Function (PRF)

A keyed function F: $\{0,1\}^{\ell_{key}(n)} \times \{0,1\}^{\ell_{in}(n)} \rightarrow \{0,1\}^{\ell_{out}(n)}$, which "looks random" without the secret key k.

- $\ell_{key}(n)$ - length of secret key k
- $\ell_{in}(n)$ - length of input
- $\ell_{out}(n)$ - length of output

- Typically, $\ell_{key}(n)$=$\ell_{in}(n)$=$\ell_{out}(n)$ =n (unless otherwise specified)

- Computing $F_K(x)$ is efficient (polynomial-time)

PRF vs. PRG

- Pseudorandom Generator G is not a keyed function
- PRG Security Model: Attacker sees only output G(r)
  - Attacker who sees r can easily distinguish G(r) from random
- PRF Security Model: Attacker sees both inputs and outputs $(r_i, F_k(r_i))$
  - In fact, attacker can select inputs r_i
  - Attacker Goal: distinguish F from a truly random function

Truly Random Function

- Let **Func$_n$** denote the set of all functions $f: \{0,1\}^n \rightarrow \{0,1\}^n$.

- **Question**: How big is the set **Func$_n$**?
- **Hint**: Consider the lookup table.
  - 2^n entries in lookup table
  - n bits per entry
  - $n2^n$ bits to encode $f \in$ **Func$_n$**
- **Answer**: $|\mathbf{Func_n}| = 2^{n2^n}$ (by comparison only 2^n n-bit keys)

Truly Random Function

- Let $\mathbf{Func_n}$ denote the set of all functions $f: \{0,1\}^n \to \{0,1\}^n$.

- Can view entries in lookup table as populated in advance (uniformly)
  - **Space:** $n2^n$ bits to encode $f \in \mathbf{Func_n}$
- Alternatively, can view entries as populated uniformly "on-the-fly"
  - **Space:** $2n \times q(n)$ bits after $q(n)$ queries
    - To store past responses

Oracle Notation

- We use $A^{f(.)}$ to denote an algorithm A with oracle access to a function f.
- A may adaptively query f(.) on multiple different inputs $x_1, x_2, \ldots$ and A receives the answers $f(x_1), f(x_2), \ldots$
- However, A can only use f(.) as a blackbox (no peaking at the source code in the box)

PRF Security

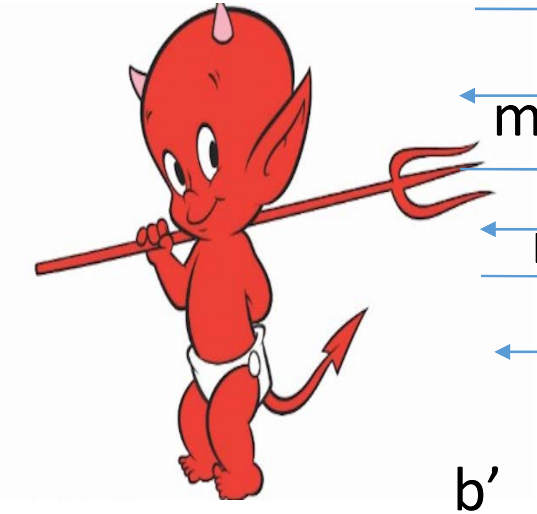
Definition 3.25: A keyed function $\mathrm{F}: \{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$ is a pseudorandom function if for all PPT distinguishers D there is a negligible function μ s.t.

$$\left|Pr\left[D^{F_k(.)}(1^n)\right] - Pr\left[D^{f(.)}(1^n)\right]\right| \leq \mu(n)$$

Notes:

- the first probability is taken over the uniform choice of $k \in \{0,1\}^n$ as well as the randomness of D.
- the second probability is taken over uniform choice of f ∈$\mathbf{Func_n}$as well as the randomness of D.
- D is *not* given the secret k in the first probability (otherwise easy to distinguish…how?)

PRF-Security as a Game



m_1 →

← $\mathbf{r_1}$

m_2 →

← $\mathbf{r_2}$

m_3 →

← $\mathbf{r_3}$

⋮

b' →

Random bit b

K = Gen(.)

Truly random func R

$r_i = F_K(m_i)$ if b=1

$R(m_i)$ o.w

$$\forall PPT\ A\ \exists \mu\ (\text{negligible})\ \text{s.t}$$

$$\Pr[A\ \ Guesses\ b' = b] \leq \frac{1}{2} + \mu(n)$$

CPA-Secure Encryption

- Gen: on input 1^n pick uniform $k \in \{0,1\}^n$
- Enc: Input $k \in \{0,1\}^n$ and $m \in \{0,1\}^n$

  Output $c = \langle r, F_k(r) \oplus m \rangle$ for uniform $r \in \{0,1\}^n$

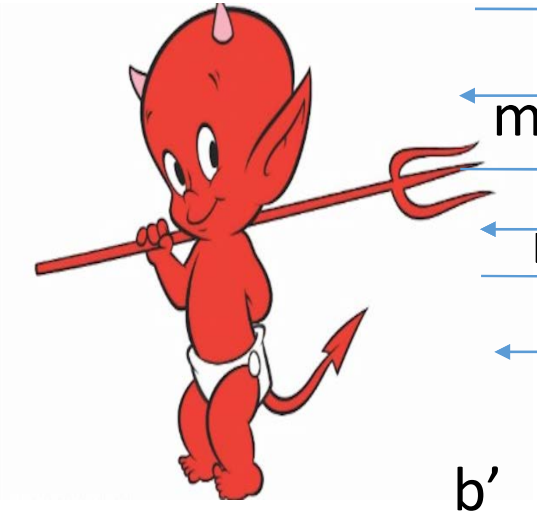
- Dec: Input $k \in \{0,1\}^n$ and $c = \langle r, s \rangle$

  Output $m = F_k(r) \oplus s$

How to begin proof?

Theorem: If F is a pseudorandom function, then (Gen,Enc,Dec) is a CPA-secure encryption scheme for messages of length n.

Breaking CPA-Security (Single Message)



m_0, m_1 →

← $\langle r, F_k(r) \oplus m_b \rangle$

m_2 →

← $\langle r, F_k(r) \oplus m_2 \rangle$

m_3 →

← $\langle r, F_k(r) \oplus m_3 \rangle$

⋮

b' →

Random bit b

K = Gen(.)

Assumption: $\exists$ PPT A, P $(\mathrm{non-negligible})$ $\mathrm{s.t}$

$$\Pr[A\ \ Guesses\ b' = b] \geq \frac{1}{2} + P(n)$$

Security Reduction

- **Step 1:** Assume for contraction that we have a PPT attacker A that breaks CPA-Security.
- **Step 2:** Construct a PPT distinguisher D which breaks PRF security.
- Distinguisher D^O (oracle O --- either f or F_k)
  - Simulate A
  - Whenever A queries its encryption oracle on a message m
    - Select random r
    - Return $c = \langle r, O(r) \oplus m \rangle$
  - Whenever A outputs messages m_0,m_1
    - Select random r and bit b
    - Return $c = \langle r, O(r) \oplus m_b \rangle$
  - Whenever A outputs b'
    - Output 1 if b=b'
    - Output 0 otherwise

Analysis: Suppose that O = f then

$$\Pr[D^{F_k} = 1] = \Pr\left[PrivK_{A,\Pi}^{cpa} = 1\right]$$

Suppose that O = f then

$$\Pr\left[D^{f} = 1\right] = \Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right]$$

where $\widetilde{\Pi}$ denotes the encryption scheme in which F_k is replaced by truly random f.

Security Reduction

- **Step 1:** Assume for contraction that we have a PPT attacker A that breaks CPA-Security.
- **Step 2:** Construct a PPT distinguisher D which breaks PRF security.
- Distinguisher D^O (oracle O --- either f or F_k)
  - Simulate A
  - Whenever A queries its encryption oracle on a message m
    - Select random r
    - Return $c = \langle r, O(r) \oplus m \rangle$
  - Whenever A outputs messages m_0,m_1
    - Select random r and bit b
    - Return $c = \langle r, O(r) \oplus m_b \rangle$
  - Whenever A outputs b'
    - Output 1 if b=b'
    - Output 0 otherwise

Analysis: Suppose that O = F_k then by PRF security, for some negligible function μ, we have

$$\left|\Pr\left[PrivK_{A,\Pi}^{cpa} = 1\right] - \Pr\left[PrivK_{A,\tilde{\Pi}}^{cpa} = 1\right]\right|$$
$$= \left|\Pr[D^{F_k} = 1] - \Pr[D^f = 1]\right| \le \mu(n)$$

Implies: $\Pr\left[PrivK_{A,\tilde{\Pi}}^{cpa} = 1\right] \ge \Pr\left[PrivK_{A,\Pi}^{cpa} = 1\right] - \mu(n)$

Security Reduction

- **Fact:** Pr $\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right] \geq$ Pr$\left[PrivK_{A,\Pi}^{cpa} = 1\right]$-$\mu(n)$

- **Claim**: For any attacker A making at most q(n) queries we have

$$\Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right] \leq \frac{1}{2} + \frac{q(n)}{2^n}$$

Conclusion: For any attacker A making at most q(n) queries we have

$$\Pr\left[PrivK_{A,\Pi}^{cpa} = 1\right] \leq \frac{1}{2} + \frac{q(n)}{2^n} + \mu(n)$$

where $\frac{q(n)}{2^n} + \mu(n)$ is negligible.

Finishing Up

Claim: For any attacker A making at most q(n) queries we have

$$\Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right] \leq \frac{1}{2} + \frac{q(n)}{2^n}$$

Proof: Let m_0,m_1 denote the challenge messages and let r* denote the random string used to produce the challenge ciphertext

$$c = \langle r^*, f(r^*) \oplus m_b \rangle$$

And let r_1,…,r_q denote the random strings used to produce the other ciphertexts $c_i = \langle r_i, f(r_i) \oplus m_b \rangle$.

If $r^* \neq r_1$,…,r_q then then c leaks no information about b (information theoretically).

Finishing Up

Claim: For any attacker A making at most q(n) queries we have

$$\Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right] \le \frac{1}{2} + \frac{q(n)}{2^n}$$

Proof: If $r^* \neq r_1,\ldots,r_q$ then then c leaks no information about b (information theoretically). We have

$$\Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1\right]$$
$$\le \Pr\left[PrivK_{A,\widetilde{\Pi}}^{cpa} = 1 \mid r* \neq r_1,\ldots,r_q\right] + \Pr[r* \in \{r_1,\ldots,rq\}\ ]$$
$$\le \frac{1}{2} + \frac{q(n)}{2^n}$$

Conclusion

$$\text{Enc}_\text{k}(\text{m}) = \langle r, F_k(r) \oplus m \rangle$$

$$\text{Dec}_\text{k}(\langle r, s \rangle) = F_k(r) \oplus s$$

PRF Security

For any attacker A making at most q(n) queries we have

$$\Pr\left[PrivK_{A,\Pi}^{cpa} = 1\right] \leq \frac{1}{2} + \frac{q(n)}{2^n} + \boxed{\mu(n)}$$

Are PRFs or PRGs more Powerful?

- Easy to construct a secure PRG from a PRF

$$G(s) = F_s(1) | \ldots | F_s(\ell)$$

- Construct a PRF from a PRG?
  - Tricky, but possible… (Katz and Lindell Section 7.5)

Construct PRF from PRG

Define: $G(s) = G_0(s) | G_1(s)$

PRF: $F_k(x) = G_{x_1}\left(\ldots G_{x_{n-1}}\left(G_{x_n}(k)\right)\right)$

Recursive Definition: $F_k(x) = H_k(x)$ where

$H_k(1) := G_1(k)$

$H_k(0) := G_0(k)$

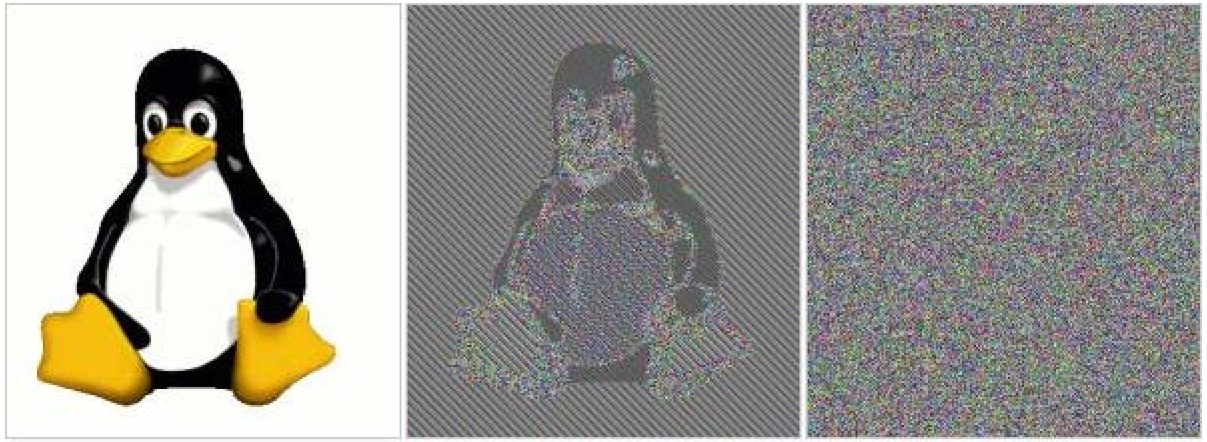
$H_k(1|x) := G_1(H_k(x))$

$H_k(0|x) := G_0(H_k(x))$

Theorem: If G is a PRG then F_k is a PRF

Next Class

- Read Katz and Lindell 3.6.2-3.6.7
- Modes of Operation
  - Stream-Cipher/Block-Cipher



Original image | Encrypted using ECB mode | Modes other than ECB result in pseudo-randomness