Cryptography
CS 555

Topic 29: Formalizing Public Key Cryptography

Recap

- Key Management
- Diffie Hellman Key Exchange
- Password Authenticated Key Exchange (PAKEs)

Public Key Encryption: Basic Terminology

- Plaintext/Plaintext Space
  - A message $\mathrm{m} \in \mathcal{M}$
- Ciphertext $\mathrm{c} \in \mathcal{C}$
- **Public/Private Key Pair $(\boldsymbol{pk}, \boldsymbol{sk}) \in \boldsymbol{\mathcal{K}}$**

Public Key Encryption Syntax

- Three Algorithms
  - $\mathrm{Gen}(1^n, R)$ (Key-generation algorithm)
    - Input: Random Bits R
    - Output: $(\boldsymbol{pk}, \boldsymbol{sk}) \in \boldsymbol{\mathcal{K}}$
  - $\mathrm{Enc}_{\mathrm{pk}}(m) \in \mathcal{C}$ (Encryption algorithm)
  - $\mathrm{Dec}_{\mathrm{sk}}(c)$ (Decryption algorithm)
    - Input: Secret key sk and a ciphertex c
    - Output: a plaintext message $\mathrm{m} \in \mathcal{M}$
- **Invariant**: $\mathrm{Dec}_{sk}(\mathrm{Enc}_{pk}(m))=m$

Alice must run key generation algorithm in advance an publishes the public key: pk

Assumption: Adversary only gets to see pk (not sk)

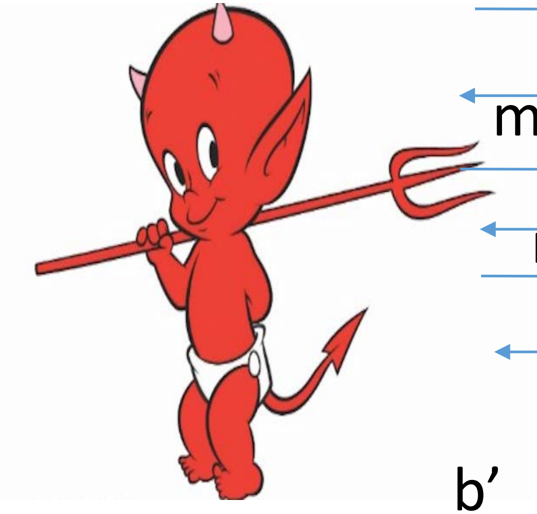
Chosen-Plaintext Attacks

- Model ability of adversary to control or influence what the honest parties encrypt.

- Historical Example: Battle of Midway (WWII).
  - US Navy cryptanalysts were able to break Japanese code by tricking Japanese navy into encrypting a particular message

- Private Key Cryptography

Recap CPA-Security (Symmetric Key Crypto)



$m_{0,1}, m_{1,1}$ →

← $c_1 = Enc_K(m_{b,1})$

$m_{0,2}, m_{1,2}$ →

← $c_2 = Enc_K(m_{b,2})$

$m_{0,3}, m_{1,3}$ →

← $c_3 = Enc_K(m_{b,3})$

⋮

b' →

Random bit b

K = Gen(.)

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

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

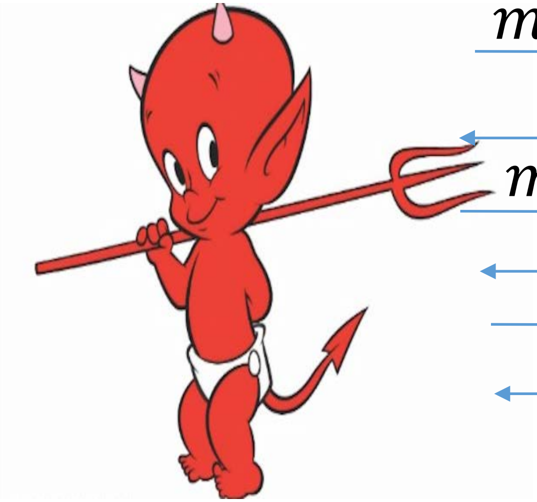
Chosen-Plaintext Attacks

- Model ability of adversary to control or influence what the honest parties encrypt.

- Private Key Crypto
  - Attacker tricks victim into encrypting particular messages

- Public Key Cryptography
  - The attacker already has the public key pk
  - Can encrypt any message s/he wants!
  - CPA Security is critical!

CPA-Security ($\mathrm{PubK}_{A,\Pi}^{\mathrm{LR-cpa}}(\mathrm{n})$)



Public Key: pk

m_0^1, m_1^1

$\boldsymbol{c_1 = \mathrm{Enc}_{pk}(m_b^1)}$

m_0^2, m_1^2

$\boldsymbol{c_2 = \mathrm{Enc}_{pk}(m_b^2)}$

m_0^3, m_1^3

$\boldsymbol{c_3 = \mathrm{Enc}_{pk}(m_b^3)}$

⋮

b'

Random bit b

(pk,sk) = Gen(.)

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

$$\Pr\left[\mathrm{PubK}_{A,\Pi}^{\mathrm{LR-cpa}}(\mathrm{n}) = 1\right] \leq \frac{1}{2} + \mu(n)$$

CPA-Security (Single Message)

$Formally, let\ \Pi = (Gen, Enc, Dec)\ denote\ the\ encryption\ scheme,$
$call\ the\ experiment\ PubK_{A,\Pi}^{LR-cpa}(n)\ and\ define\ a\ random\ variable$

$$\mathrm{PubK}_{\mathrm{A},\Pi}^{\mathrm{LR-cpa}}(n) = 1 \quad if\ b = b'$$
$$\mathrm{PubK}_{\mathrm{A},\Pi}^{\mathrm{LR-cpa}}(n) = 0 \quad otherwise$$

$\Pi\ has\ indistinguishable\ encryptions\ under\ a\ chosen\ plaintext\ attack$
$if\ for\ all\ PPT\ adversaries\ A, there\ is\ a$ negligible function μ such that

$$\Pr[\mathrm{PubK}_{\mathrm{A},\Pi}^{\mathrm{LR-cpa}}(n) = 1] \leq \frac{1}{2} + \mu(n)$$

Private Key Crypto

- CPA Security was stronger than eavesdropping security

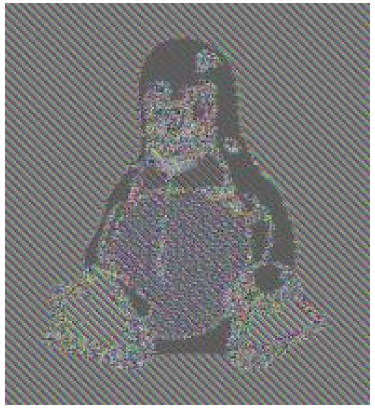
$$\mathrm{Enc}_{\mathrm{K}}(\mathrm{m}) = \mathrm{G}(\mathrm{K}) \oplus m$$

Vs.

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

Public Key Crypto

- **Fact 1:** CPA Security and Eavesdropping Security are Equivalent
  - Key Insight: The attacker has the public key so he doesn't gain anything from being able to query the encryption oracle!
- **Fact 2:** Any deterministic encryption scheme is not CPA-Secure
  - Historically overlooked in many real world public key crypto systems
- **Fact 3:** Plain RSA is not CPA-Secure
- **Fact 4:** No Public Key Cryptosystem can achieve Perfect Secrecy!
  - Exercise 11.1
  - Hint: Unbounded attacker can keep encrypting the message m using the public key to recover all possible encryptions of m.



Encrypting Longer Messages

Claim 11.7: Let $\Pi = (Gen, Enc, Dec)$ denote a CPA-Secure public key encryption scheme and let $\Pi' = (Gen, Enc', Dec')$ be defined such that

$$\mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_1} \parallel \boldsymbol{m_2} \parallel \cdots \parallel \boldsymbol{m_\ell}) = \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_1}) \parallel \cdots \parallel \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_\ell})$$

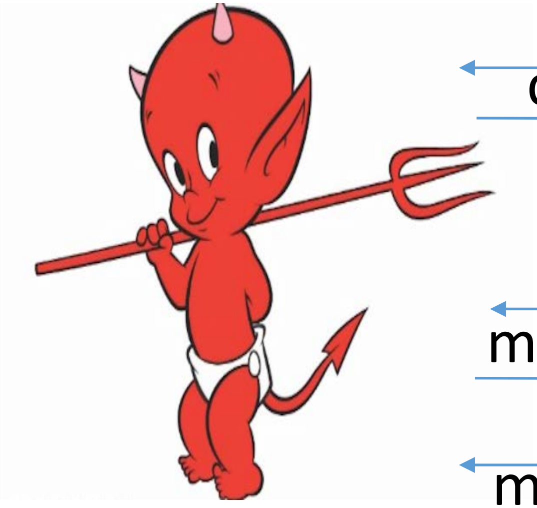
Then Π' is also CPA-Secure.

Chosen Ciphertext Attacks

- Models ability of attacker to obtain (partial) decryption of selected ciphertexts

- Attacker might intercept ciphertext c (sent from S to R) and send c’ instead.
  - After that attacker can observe receiver’s behavior (abort, reply etc…)
- Attacker might send a modified ciphertext c’ to receiver R in his own name.
  - E-mail response: Receiver might decrypt c’ to obtain m’ and include m’ in the response to the attacker

Recap CCA-Security (Symmet



m_{-1}

$c_{-1} = Enc_K($

c_{-2}

$m_{-2} = Dec_K(c_{-2})$

⋮

m_0, m_1

$c = Enc_K(m_b)$

m_3

$c_2 = Enc_K(m_2)$

c_3

$m_3 = Dec_K(m_3)$

$c_4 = c$

⋮

"No Way!"

b'

We could set $m_0 = m_{-1}$ or $m_1 = m_{-2}$

However, we could still flip 1 bit of c and ask challenger to decrypt

Random bit b

K = Gen(.)

Recap CCA-Security $\left(PrivK_{A,\Pi}^{cca}(n)\right)$

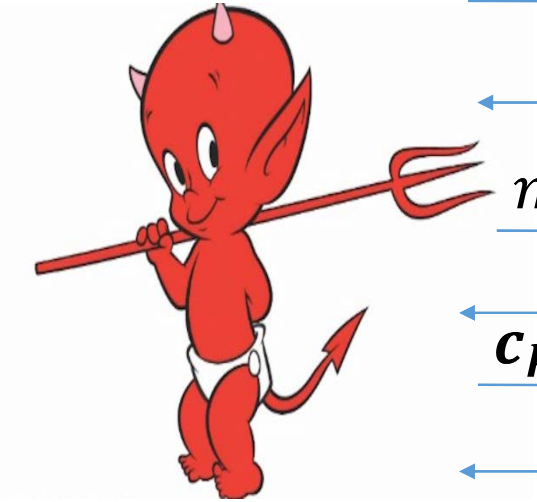
1. Challenger generates a secret key k and a bit b
2. Adversary (A) is given oracle access to Enc_k and Dec_k
3. Adversary outputs m_0,m_1
4. Challenger sends the adversary $c=Enc_k(m_b)$.
5. Adversary maintains oracle access to Enc_k and Dec_k ,however the adversary is not allowed to query $Dec_k(c)$.
6. Eventually, Adversary outputs b'.

$$PrivK_{A,\Pi}^{cca}(n) = 1 \text{ if } \mathrm{b} = \mathrm{b}'; \text{otherwise } 0.$$

CCA-Security: For all PPT A exists a negligible function negl(n) s.t.

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

CCA-Security ($\mathrm{PubK}^{\mathrm{cca}}_{A,\Pi}(n)$)



Public Key: pk

c_{-1}

$m_{-1} = \mathbf{Dec}_{sk}(c_{-1})$

⋮

m_0, m_1

$c_b = \mathbf{Enc}_{pk}(m_b)$

c_k

$m_k = \mathbf{Dec}_{sk}(c_k)$

b'

⋮

Random bit b

(pk,sk) = Gen(.)

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

$$\Pr\left[\mathrm{PubK}^{\mathrm{cca}}_{A,\Pi}(n) = 1\right] \leq \frac{1}{2} + \mu(n)$$

Encrypting Longer Messages

Claim 11.7: Let $\Pi = (Gen, Enc, Dec)$ denote a CPA-Secure public key encryption scheme and let $\Pi' = (Gen, Enc', Dec')$ be defined such that

$$\mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_1} \parallel \boldsymbol{m_2} \parallel \cdots \parallel \boldsymbol{m_\ell}) = \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_1}) \parallel \cdots \parallel \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_\ell})$$

Then Π' is also CPA-Secure.

Claim? Let $\Pi = (Gen, Enc, Dec)$ denote a CCA-Secure public key encryption scheme and let $\Pi' = (Gen, Enc', Dec')$ be defined such that

$$\mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_1} \parallel \boldsymbol{m_2} \parallel \cdots \parallel \boldsymbol{m_\ell}) = \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_1}) \parallel \cdots \parallel \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_\ell})$$

Then Π' is also CCA-Secure.

Is this second claim true?

Encrypting Longer Messages

Claim? Let $\Pi = (Gen, Enc, Dec)$ denote a CCA-Secure public key encryption scheme and let $\Pi' = (Gen, Enc', Dec')$ be defined such that

$$\mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_1} \parallel \boldsymbol{m_2} \parallel \cdots \parallel \boldsymbol{m_\ell}) = \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_1}) \parallel \cdots \parallel \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_\ell})$$

Then Π' is also CCA-Secure.

Is this second claim true?

Answer: No!

Encrypting Longer Messages

Fact: Let $\Pi = (Gen, Enc, Dec)$ denote a CCA-Secure public key encryption scheme and let $\Pi' = (Gen, Enc', Dec')$ be defined such that

$$\mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_1} \parallel \boldsymbol{m_2} \parallel \cdots \parallel \boldsymbol{m_\ell}) = \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_1}) \parallel \cdots \parallel \mathbf{Enc}_{\mathbf{pk}}(\boldsymbol{m_\ell})$$

Then Π' is **Provably Not** CCA-Secure.

1. Attacker sets $\boldsymbol{m_0} = \mathbf{0}^n \parallel \mathbf{1}^n \parallel \mathbf{1}^n$ and $\boldsymbol{m_1} = \mathbf{0}^n \parallel \mathbf{0}^n \parallel \mathbf{1}^n$ and gets $\boldsymbol{c_b} = \mathbf{Enc}'_{\mathbf{pk}}(\boldsymbol{m_b}) = \boldsymbol{c_{b,1}} \parallel \boldsymbol{c_{b,2}} \parallel \boldsymbol{c_{b,3}}$
2. Attacker sets $\boldsymbol{c'} = \boldsymbol{c_{b,2}} \parallel \boldsymbol{c_{b,3}} \parallel \boldsymbol{c_{b,1}}$, queries the decryption oracle and gets

$$\mathbf{Dec}'_{\mathbf{sk}}(\boldsymbol{c'}) = \begin{cases} \mathbf{1}^n \parallel \mathbf{1}^n \parallel \mathbf{0}^n & \textbf{if b=0} \\ \mathbf{0}^n \parallel \mathbf{1}^n \parallel \mathbf{0}^n & \boldsymbol{otherwise} \end{cases}$$

Achieving CPA and CCA-Security

- Plain RSA is not CPA Secure (therefore, not CCA-Secure)

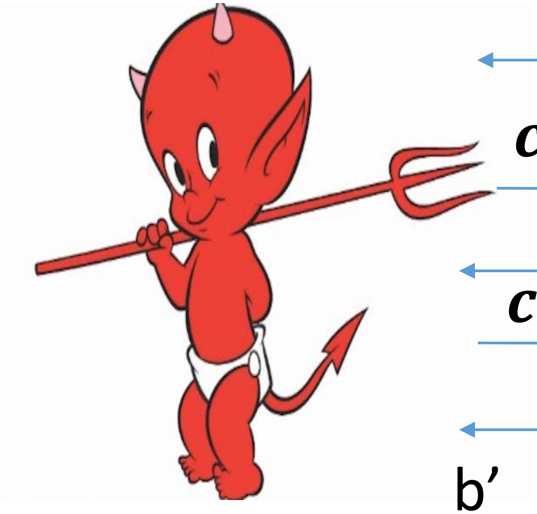
- El-Gamal (next class) is CPA-Secure, but not CCA-Secure
  - Homework 4

- Tools to build CCA-Secure Encryption
  - Key Encapsulation Mechanism

Key Encapsulation Mechanism (KEM)

- Three Algorithms
  - $\mathrm{Gen}(1^n, R)$ (Key-generation algorithm)
    - Input: Random Bits R
    - Output: $(\boldsymbol{pk}, \boldsymbol{sk}) \in \boldsymbol{\mathcal{K}}$
  - $\mathrm{Encaps}_{\mathrm{pk}}(1^n, R)$
    - Input: security parameter, random bits R
    - Output: Symmetric key $\mathrm{k} \in \{0,1\}^{\ell(n)}$ and a ciphertext c
  - $\mathrm{Decaps}_{\mathrm{sk}}(c)$ (Deterministic algorithm)
    - Input: Secret key $\mathrm{sk} \in \mathcal{K}$ and a ciphertex c
    - Output: a symmetric key$\{0,1\}^{\ell(n)}$ or $\perp$ (fail)
- **Invariant**: $\mathrm{Decaps}_{\mathrm{sk}}$(c)=k whenever (c,k) = $\mathrm{Encaps}_{\mathrm{pk}}(1^n, R)$

KEM CCA-Security ($\mathrm{KEM}^{\mathrm{cca}}_{\mathrm{A},\Pi}(\mathrm{n})$)



$(\boldsymbol{pk}, \boldsymbol{c}, \boldsymbol{k_b})$

⋮

$\boldsymbol{c_1 \neq c}$

$\mathbf{Decaps}_{\boldsymbol{sk}}(\boldsymbol{c_1})$

$\boldsymbol{c_2 \neq c}$

$\mathbf{Decaps}_{\boldsymbol{sk}}(\boldsymbol{c_2})$

b'

⋮

Random bit b

(pk,sk) = Gen(.)

$\boldsymbol{(c, k_0) = \mathbf{Encaps}_{pk}(.)}$

$\boldsymbol{k_1 \leftarrow \{0,1\}^n}$

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

$$\Pr\left[\mathrm{KEM}^{\mathrm{cca}}_{\mathrm{A},\Pi} = 1\right] \leq \frac{1}{2} + \mu(n)$$

CCA-Secure Encryption from CCA-Secure KEM

$$\mathbf{Enc_{pk}}(\boldsymbol{m}; \boldsymbol{R}) = \langle \boldsymbol{c}, \mathbf{Enc}^*_{\mathbf{k}}(\boldsymbol{m}) \rangle$$

Where

- $(\boldsymbol{c}, \boldsymbol{k}) \leftarrow \mathbf{Encaps_{pk}}(\mathbf{1}^{\boldsymbol{n}}; \boldsymbol{R})$,
- $\mathbf{Enc}^*_{\mathbf{k}}$ is a CCA-Secure symmetric key encryption algorithm, and
- $\mathbf{Encaps_{pk}}$ is a CCA-Secure KEM.

Theorem 11.14: $\mathbf{Enc_{pk}}$ is CCA-Secure public key encryption scheme.

CCA-Secure KEM in the Random Oracle Model

- Let (N,e,d) be an RSA key (pk =(N,e), sk=(N,d)).

$$\mathrm{Encaps}_{\mathrm{pk}}(1^n, R) = \left(r^e \; mod \; N, k = H(r)\right)$$

- Remark 1: k is completely random string unless the adversary can query random oracle H on input r.
- Remark 2: If Plain-RSA is hard to invert for a random input then PPT attacker finds r with negligible probability.

Next Class: El-Gamal

- Read Katz and Lindell: 11.4