Secure

```
Enc(k, m):
  r ← {0,1}^λ
  c ← F(k, r) ⊕ m
  return (c, r)
```

Insecure

```
Enc(k, m):
  r ← {0,1}^λ
  c = F(k, m) ⊕ r
  return (c, r)
```

Attack

$F(k, m) = c \oplus r$

```
A
c0, r0 ← Query(0^λ, 1^λ)
c1, r1 ← Query(0^λ, 0^λ)
if c0 ⊕ r0 = c1 ⊕ r1:
    return 1          "left messages"
else
    return 0
```

- Randomization alone is not sufficient for CPA security (is necessary though)
-

CPA vs. CPA\$

$\mathcal{L}_{\text{CPA\$-real}}$

$$\boxed{\begin{array}{l} K \leftarrow \Sigma.\mathcal{K} \\ \\ \underline{\text{CHALLENGE}(m):} \\ \quad \text{return } \Sigma.\text{Enc}(K, m) \end{array}}$$

$\mathcal{L}_{\text{CPA\$-rand}}$

$$\boxed{\begin{array}{l} \underline{\text{CHALLENGE}(m):} \\ \quad c \leftarrow \{0,1\}^{2\lambda} \\ \quad \text{return } c \end{array}}$$

- CPA\$ requires uniformly random ciphertexts
- CPA ciphertexts may have special format...

CPA\$ implies CPA

CPA\$ means $\boxed{c \leftarrow \text{Enc}(K, m)} \approx \boxed{c \leftarrow \{0,1\}^{2\lambda}}$

$$\boxed{\begin{array}{l} \underline{\text{Query}(m_0, m_1):} \\ \quad \text{return Enc}(K, m_0) \end{array}} \approx \boxed{\begin{array}{l} \underline{\text{Query}} \\ \quad c \leftarrow \{0,1\}^{2\lambda} \\ \quad \text{return } c \end{array}}$$

$$\approx \boxed{\begin{array}{l} \underline{\text{Query}(m_0, m_1):} \\ \quad \text{return Enc}(K, m_1) \end{array}}$$

CPA does NOT imply CPA\$

- let Σ be a CPA\$

$$\boxed{\begin{array}{l} \underline{\text{Enc}'(K, m):} \\ \quad \text{return } \Sigma.\text{Enc}(K, m) \| 00 \end{array}}$$

$$\boxed{\begin{array}{l} \text{Dec}(K, C): \\ \quad \text{throw away last 2 bits of } C \\ \quad \text{and decrypt with } \Sigma. \end{array}}$$

- Easy to show CPA secure
  - Adding zeros does not reveal any information

- Enc′ ciphertexts are not uniformly distributed.

Secure / Insecure ?

Enc(K, m):
$r \leftarrow \{0,1\}^\lambda$
$x := F(K, r \oplus m)$
return (r, x)

$\equiv$

Enc
$r' \leftarrow \{0,1\}^\lambda$
$r := m \oplus r'$
$x := F(K, r')$
return $(r, x) = (m \oplus r', F(K, r'))$

$\approx$

$r' \leftarrow$ samp() // sample w/o replacement
$r := m \oplus r'$
$x \leftarrow \{0,1\}^\lambda$ // since r' is distinct
return $(m \oplus r', x)$

~~$\equiv$~~

~~$r' \leftarrow \{0,1\}^\lambda$~~
~~$r'' \leftarrow$ samp()~~
~~return (r'', r')~~

// ~~One time pad~~ ~~rule~~

// sample w/ replacement

$\approx$

$r' \leftarrow \{0,1\}^\lambda$
$r := m \oplus r'$
$x \leftarrow \{0,1\}^\lambda$
return $(m \oplus r', x)$

$\equiv$

$c \leftarrow \{0,1\}^\lambda$
$x \leftarrow \{0,1\}^\lambda$
return (c, x)

// one time pad rule

Secure / Insecure ?

Enc(k, m): $r \leftarrow \{0,1\}^\lambda$ $x := r \oplus m$ $y := F(k, r)$ $z := F(k, x)$ return (y, z)	$\equiv$	Enc( ): $r \leftarrow \{0,1\}^\lambda$ $y := F(k, r)$ $z := F(k, r \oplus m)$ return (y, z)

- Observe, when $m = 0^\lambda$ we get

$$y = F(k, r)$$
$$z = F(k, r \oplus m) = F(k, r)$$
$$\Rightarrow y = z \text{ if } m = 0^\lambda$$

A

```
m_0 = 0^λ
m_1 = 1^λ
(y, z) = CHALLENGE(m_0, m_1)

if y = z:
    return 0
else
    return 1
```