

d) $F'(K, n) = F(K, 0^n) \circ f(K, n)$

$F'(K, n)$ is not secure.

$$f^1: K \rightarrow 2K \mid \begin{array}{c} \overline{m_1 \parallel m_2} \\ \text{first } K^{\text{th}} \text{ bits} \quad \text{last } K^{\text{th}} \text{ bits} \end{array}$$

f^1 - real

\Rightarrow
 $\begin{array}{l} \text{A} \\ m_1 \parallel m_2 := f'_K(n_1) \\ m_3 \parallel m_4 := f'_K(n_2) \\ \text{return}(m_1 \parallel m_3) \end{array}$
 \diamond

$\begin{array}{l} K \leftarrow \{0, 1\}^K \\ \underline{f'_K(n)} \\ \text{return}(f(K, 0^n) \parallel f(K, n)) \end{array}$

As $F(K, 0^n)$ will always spit the same bits. Hence we can query two times and see if our $\text{first } K^{\text{th}} \text{ bits}$ from $(m_1 \parallel m_2)$ match the first K^{th} bits from $(m_3 \parallel m_4)$

This would not be possible if we were querying a real pseudorandom function.

$$\Pr[A \triangleleft f^1\text{-real} \rightarrow 1] - \Pr[A \triangleleft \overbrace{f^1\text{-rand}}^{\text{just a doubling PRF}} \rightarrow 1]$$

$$\Rightarrow 1 - \frac{1}{2^K} \leftarrow \text{negligible}$$

\uparrow
big

It is non-negligible

15

$$F^2(K, n) = F(K, n) \circ F(K, \bar{n})$$

$F^2(K, n)$ is a secure PRF.

Given $F \approx \text{prf-real}$

To prove $F^2 \approx \text{prf-real}$

$F^2 \approx \text{prf-real}$

```

K ← {0, 1}^n
Query(x ∈ {0, 1}^n)
return (F(K, x) || F(K, x̄))
    
```

AC):

```

return
(Query'(x) || Query'(x̄))
    
```

Just Moving Code

$F \approx \text{prf-real}$

```

K ← {0, 1}^n
Query'(x ∈ {0, 1}^n)
return (F(K, x))
    
```

$F \approx \text{prf-real}$

AC):

```

return
(Query'(x) || Query'(x̄))
    
```

AC):

```

T := empty array
Query'(x ∈ {0, 1}^n)
if T[x] undefined
    T[x] ← {0, 1}^out
return T[x]
    
```

Given $F \approx \text{prf-real}$

Just moving code

AC):

```

T := empty array
Query(x ∈ {0, 1}^n):
    if T[x] undefined:
        a ← {0, 1}^out
        b ← {0, 1}^out
        T[x] := a || b
    return T[x]
    
```

$F^2 \approx \text{prf-real}$

AC):

```

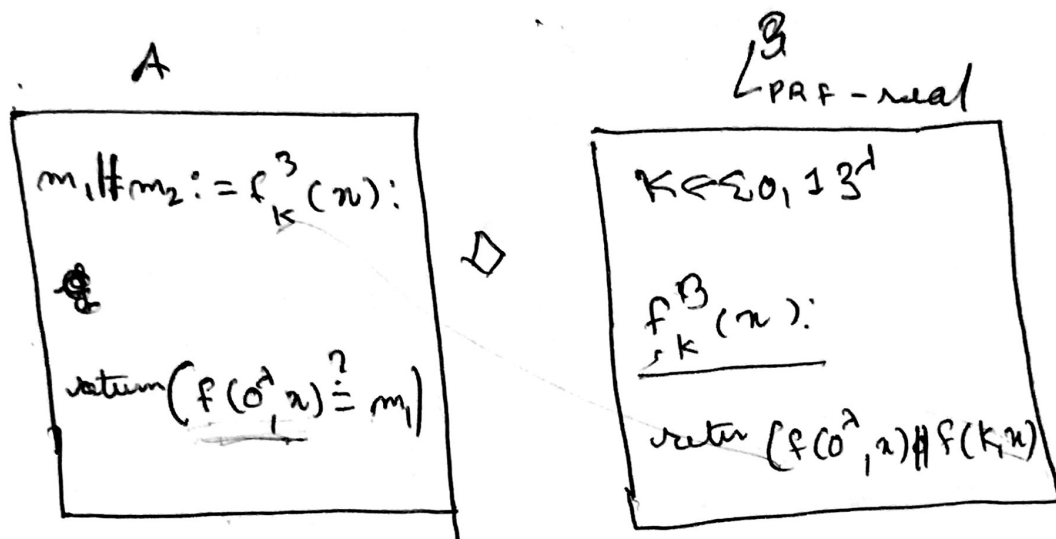
T := empty array
Query(x ∈ {0, 1}^n):
    if T[x] undefined
        T[x] ← {0, 1}^2out
    return T[x]
    
```

(C)

$$f^3(K, n) = f(O^1, n) \circ f(K, n)$$

Not a PRF

The key is O^1 is known by the adversary. Hence $f(O^1, n)$ is not a pseudorandom function. K needs to be sampled from a uniform keyspace to make our pseudorandom



the Adversary knows the PRF $f(O^1, n)$ he can easily distinguish $L_{\text{PRF-real}}^3$ from $L_{\text{PRF-random}}^3$.

The advantage is non negligible.

$$\Pr[A \rightarrow L_{\text{PRF-real}}^3 \rightarrow 1] - \Pr[A \rightarrow L_{\text{PRF-random}}^3 \rightarrow 1]$$

$$\Rightarrow \frac{1}{\uparrow \text{not negl}} - \frac{1}{2^k} \quad \text{where } \frac{1}{2^k} \uparrow \text{negl}$$

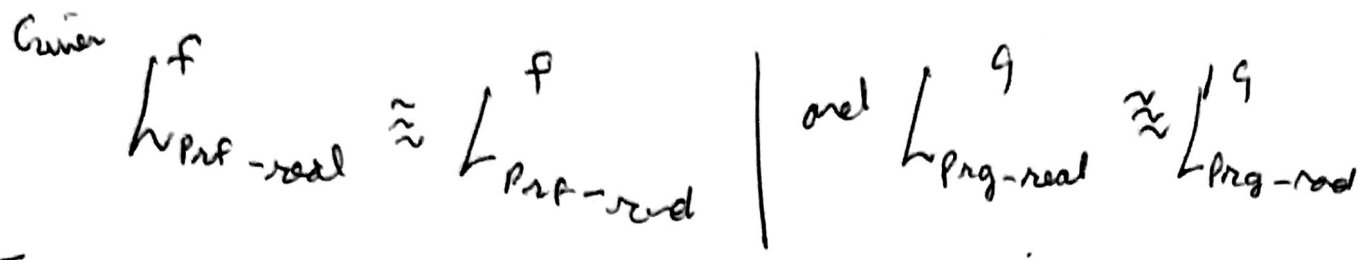
where $|m_1| = |m_2|$

Hence this advantage is negligible.

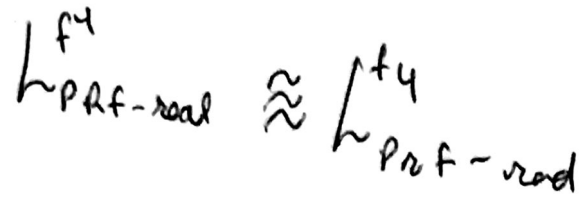
(d)

$$F^4(k, n) = G(F(k, n))$$

This is a secure pseudorandom function



To prove



$\downarrow F^4$
PRF-real

```

AC ):
  K ← {0, 1}^n
  query(x ∈ {0, 1}^n)
  return( G(F(K, x)) )
  
```

\approx

```

AC ):
  return
  G(Query'(x))
  
```

\diamond

$\downarrow F$
PRF-real

```

  K ← {0, 1}^n
  Query'(x ∈ {0, 1}^n)
  return(F(K, x))
  
```

\approx

```

AC ):
  return
  (G(Query'(x)))
  
```

\diamond

$\downarrow F$
PRF-real

```

T := empty array
Query'(x ∈ {0, 1}^n)
if T[x] undefined
  T[x] ← {0, 1}^n
return T[x]
  
```

Given
 $\downarrow F$
PRF-real
 $\downarrow F$
PRF-real

\approx

```

T := empty array
query(x ∈ {0, 1}^n)
if T[x] undefined
  a ← {0, 1}^n
  T[x] ← G(a)
return( T[x] )
  
```

α is uniformly random
So $G(\alpha)$ can double
it and also we know
 $G(x)$ is a PRG
which can be replaced
with $\downarrow F$
PRG-real

~
~

```

T := empty array
query( $x \in \{0, 1\}^n$ )
if  $T[x]$  undefined
     $T[x] := query'(x)$ 
return( $T[x]$ )
    
```

↔

\hookrightarrow $L_{prog-real}$

```

query'(c)
 $a \leftarrow \{0, 1\}^{2out}$ 
return  $q(a)$ 
    
```

moving the code

~
~

```

T := empty array
query( $x \in \{0, 1\}^n$ )
if  $T[x]$  undefined:
     $T[x] := query'(x)$ 
return( $T[x]$ )
    
```

↔

\hookrightarrow $L_{prog-real}$

```

query'(x):
 $z \leftarrow \{0, 1\}^{2out}$ 
return  $z$ 
    
```

q
as $L_{prog-real} \approx$
 q
 $L_{prog-real}$

~
~

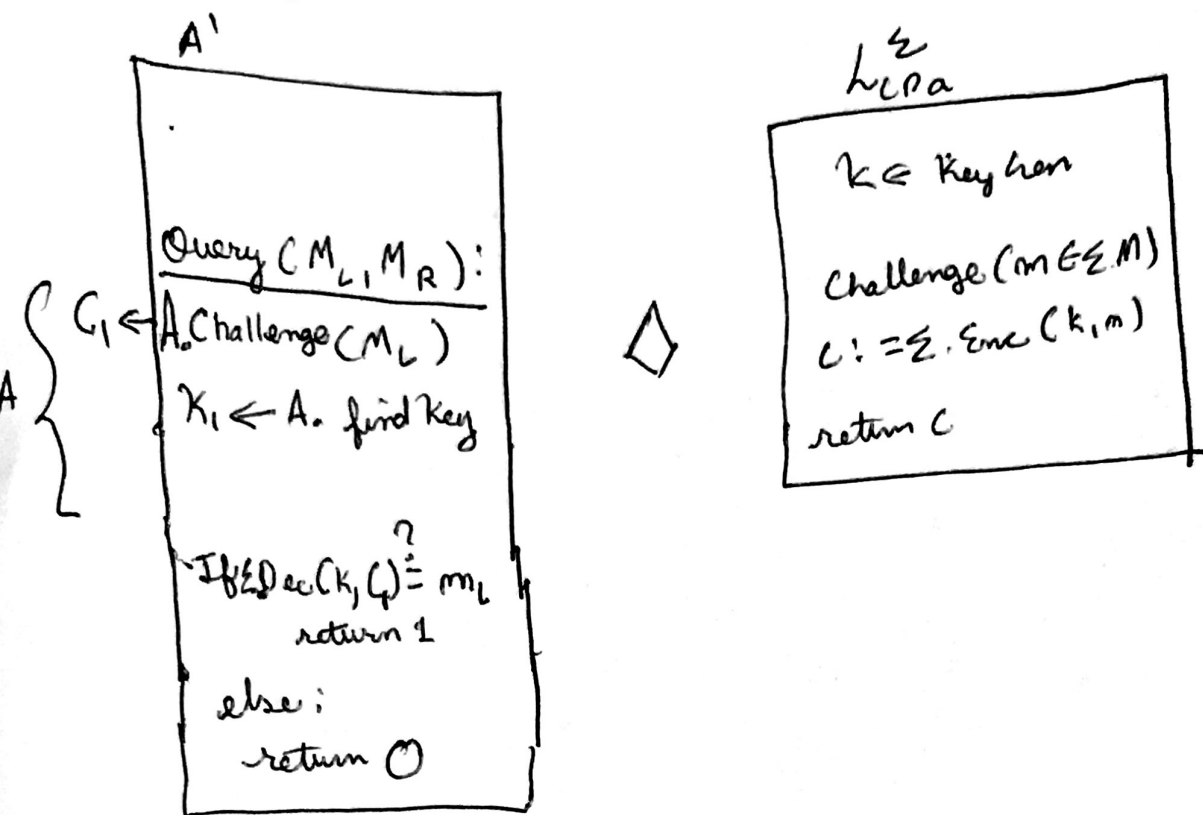
\hookrightarrow $L_{PRF-real}$

```

T := empty array
query( $x \in \{0, 1\}^n$ )
if  $T[x]$  undefined:
     $T[x] \leftarrow \{0, 1\}^{2out}$ 
return( $T[x]$ )
    
```

Hence \hookrightarrow $L_{PRF-real} \approx$ \hookrightarrow $L_{PRF-real}$

2) Problem



In the m_L world we would always return 1 whereas an m_R world our adversary may not return 1. It returns 1 with negligible probability

Hence we can distinguish $LCPA_{\text{left}}$ from $LCPA_{\text{right}}$