One-time Semantic Security and Pseudorandom Functions

CS/ECE 407

Today's objectives

See an attack on a "PRG"

Use PRG to define a new cipher

Define interchangeability

Define one-time semantic security

Prove our cipher satisfies one-time semantic security

Introduce Pseudorandom Functions (PRFs)

Indistinguishability

$$L \overset{c}{\approx} R$$

Two programs L and R are **indistinguishable** if for *any* polynomial-time program A outputting a bit, the following quantity is negligible (in λ):

$$|\Pr[\ A \diamond L = 1\ ] - \Pr[\ A \diamond R = 1\ ]|$$

Indistinguishability

$$L \overset{c}{\approx} R$$

Two programs L and R are **indistinguishable** if for *any* polynomial-time program A outputting a bit, the following quantity is negligible (in λ):

$$|\Pr[\ A \diamond L = 1\ ] - \Pr[\ A \diamond R = 1\ ]|$$

Pseudorandom Generators

G is a PRG if the following indistinguishability holds:

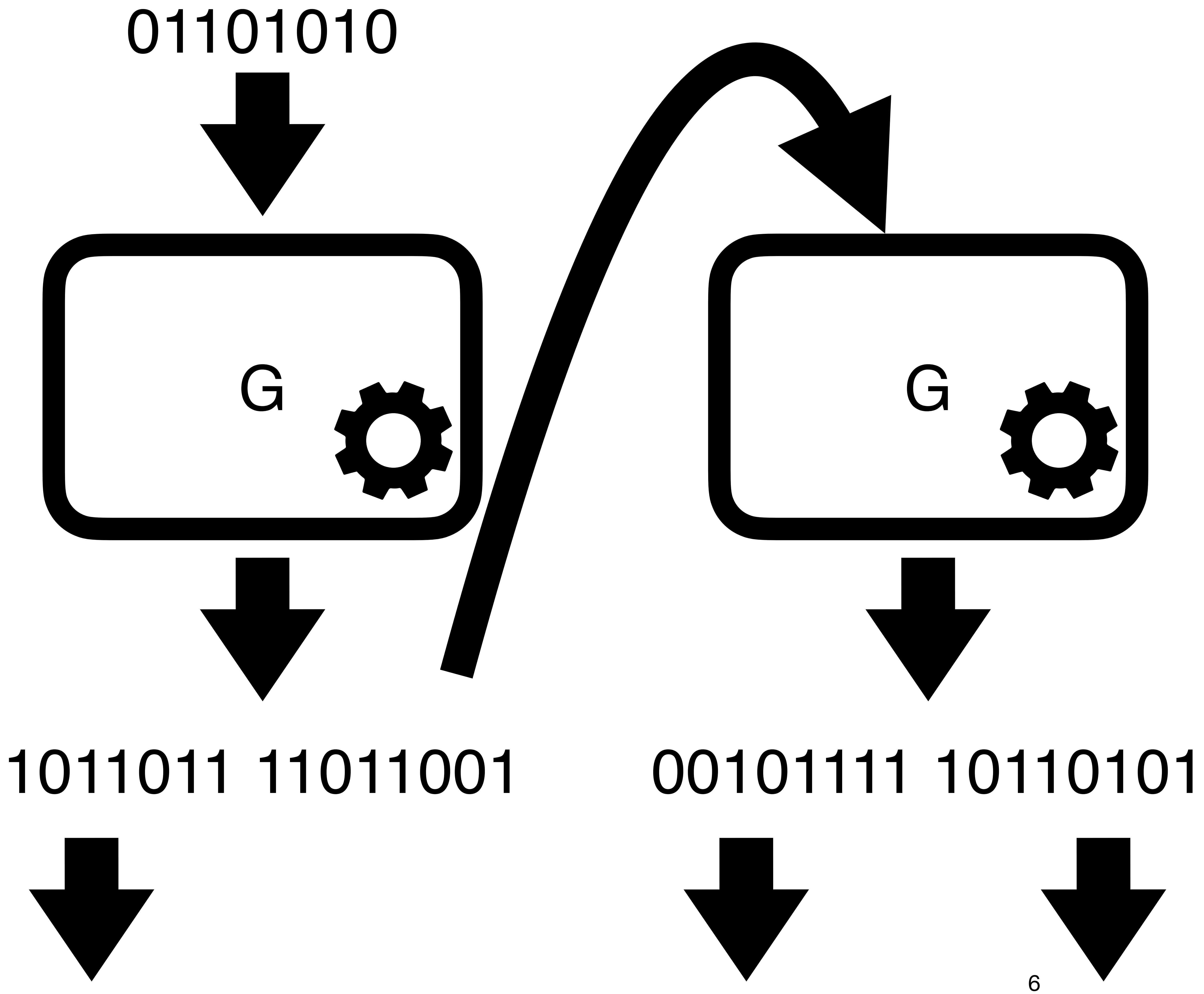
```
gen():
  seed ←$ {0,1}^n
  return G(seed)
```

$\overset{c}{\approx}$

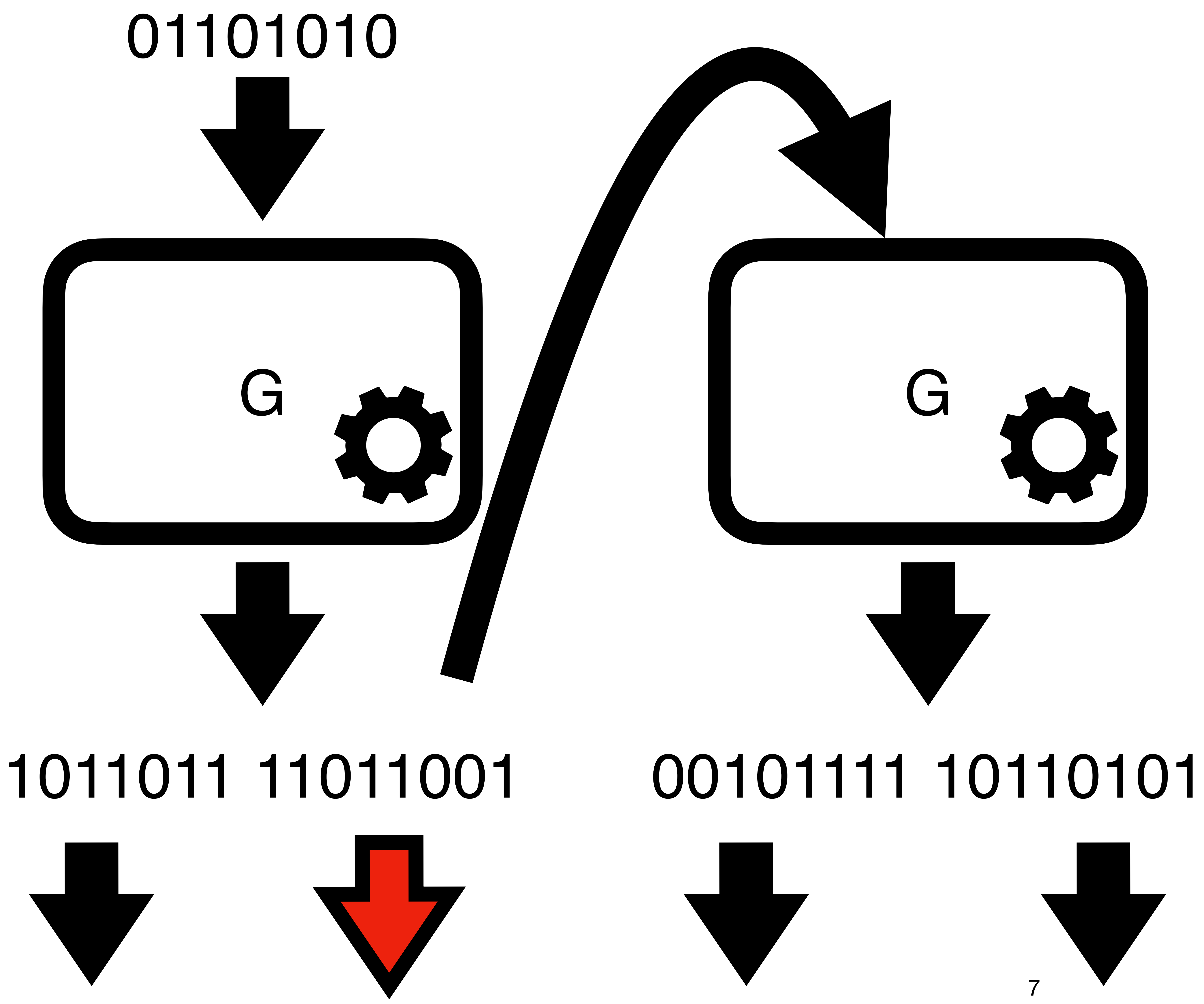
```
gen():
  r ←$ {0,1}^(n+s)
  return r
```

There is no PPT program that can distinguish the above two programs by just calling them

Stretching the output of a PRG

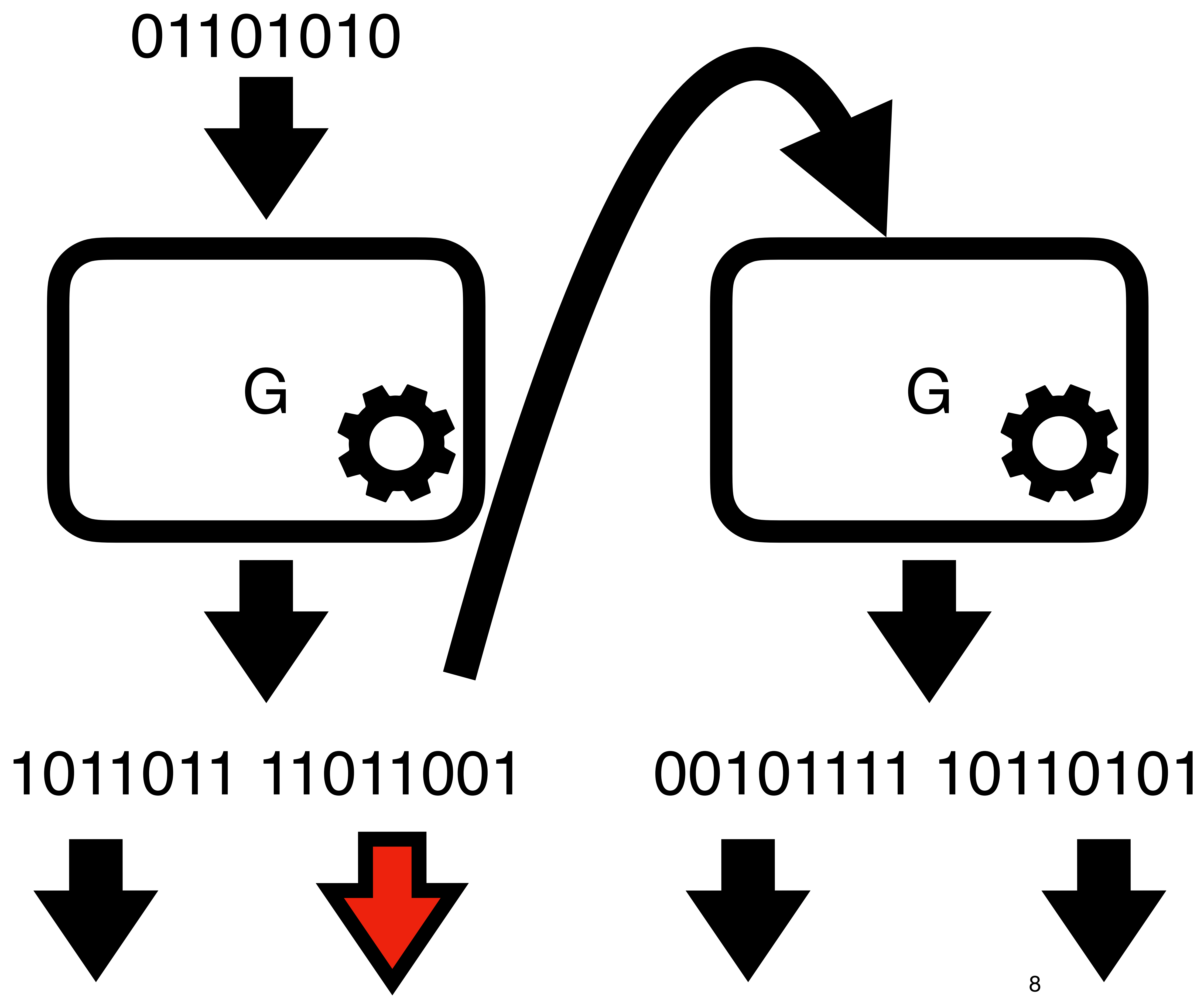


Stretching the output of a PRG

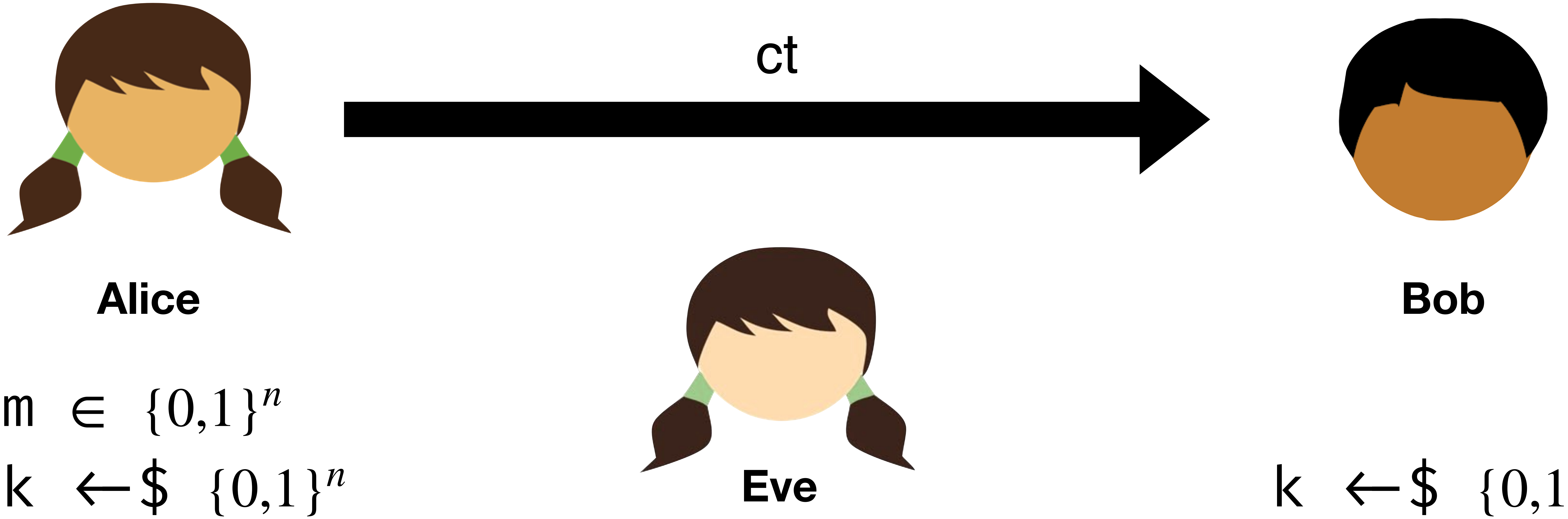


Is this secure?

Stretching the output of a PRG




```
G'():
  s ←$ {0,1}^λ
  w||x ← G(s)
  y||z ← G(x)
  return w||x||y||z
```



Alice

$m \in \{0,1\}^n$

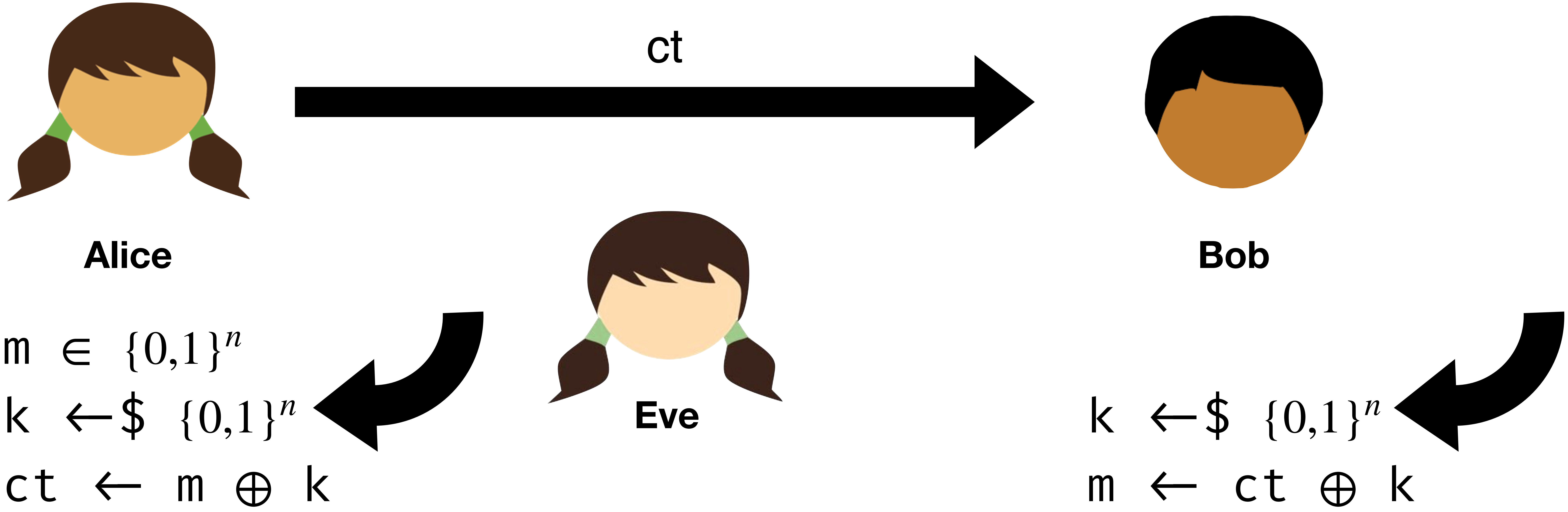
$k \xleftarrow{\$} \{0,1\}^n$

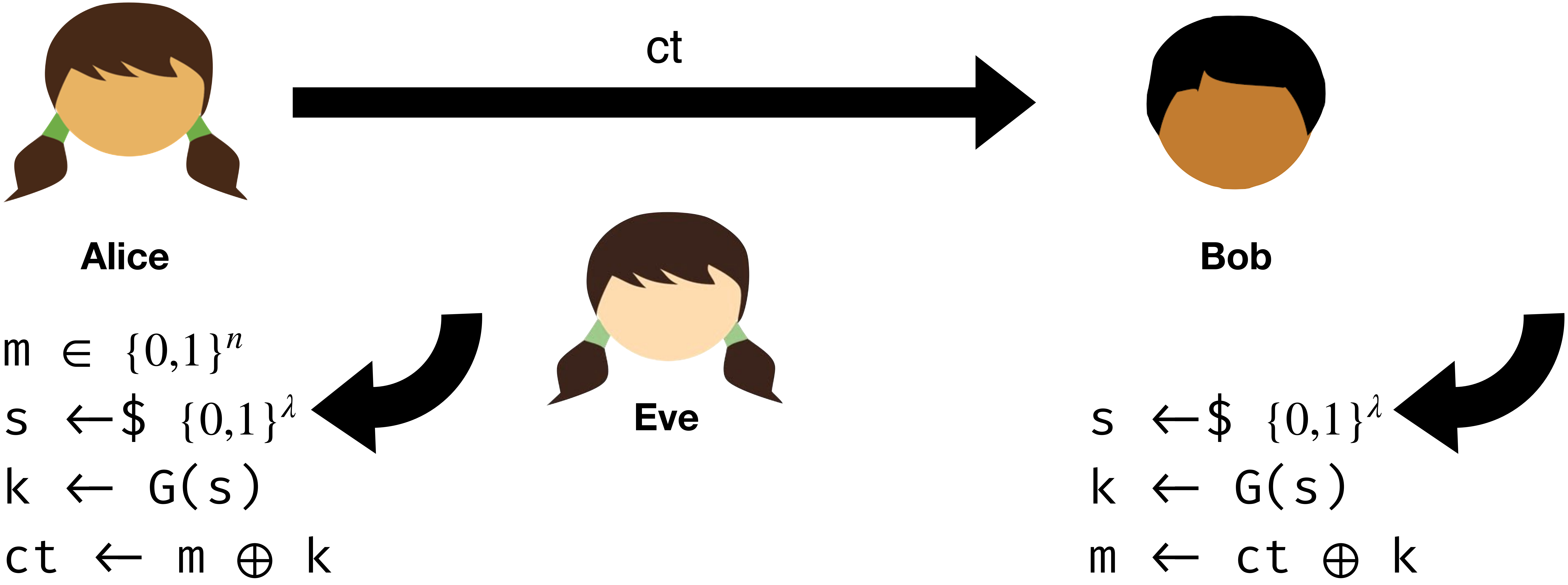
$ct \leftarrow m \oplus k$

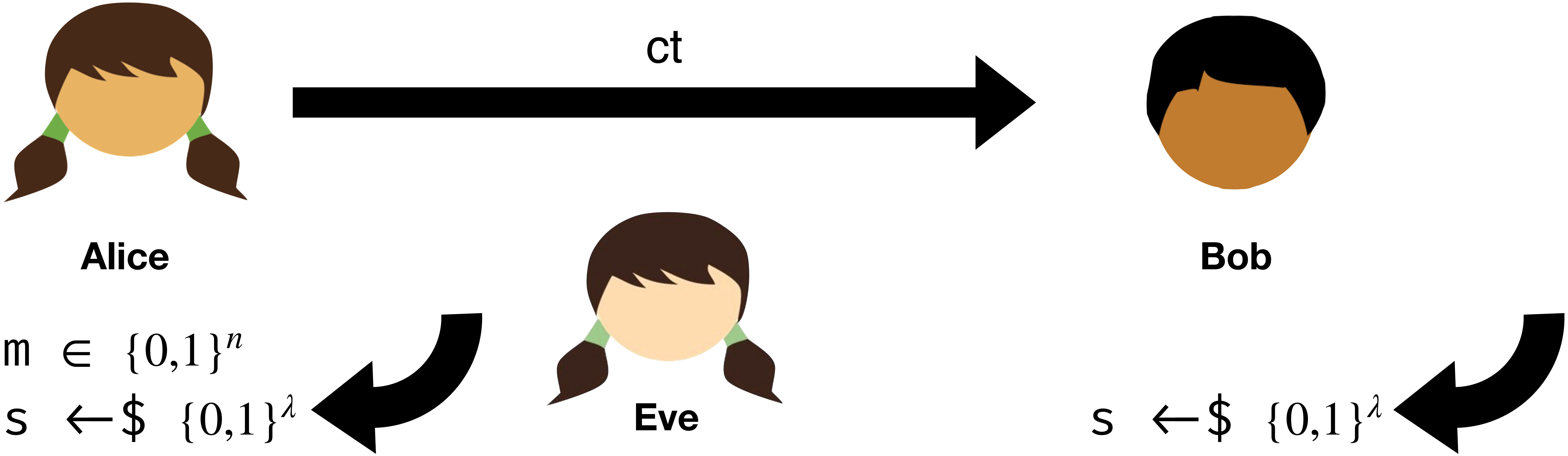
Bob

$k \xleftarrow{\$} \{0,1\}^n$

$m \leftarrow ct \oplus k$


ct
Alice
Eve
Bob
m ∈ {0,1}^n
k ←$ {0,1}^n
ct ← m ⊕ k
k ←$ {0,1}^n
m ← ct ⊕ k


ct
Alice
Eve
Bob
m ∈ $\{0,1\}^n$
s ←$ $\{0,1\}^\lambda$
k ← G(s)
ct ← m ⊕ k
s ←$ $\{0,1\}^\lambda$
k ← G(s)
m ← ct ⊕ k



Alice:

$m \in \{0,1\}^n$

$s \leftarrow\$ \{0,1\}^\lambda$

$k \leftarrow G(s)$

$ct \leftarrow m \oplus k$

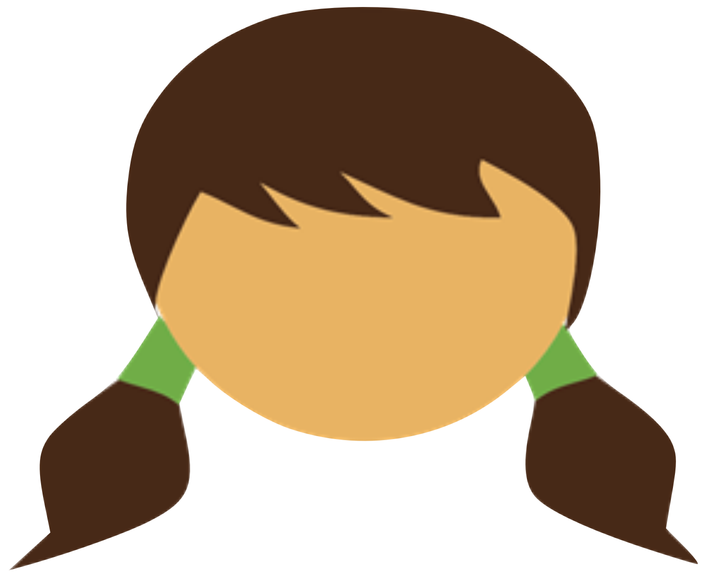
Bob:

$s \leftarrow\$ \{0,1\}^\lambda$

$k \leftarrow G(s)$

$m \leftarrow ct \oplus k$

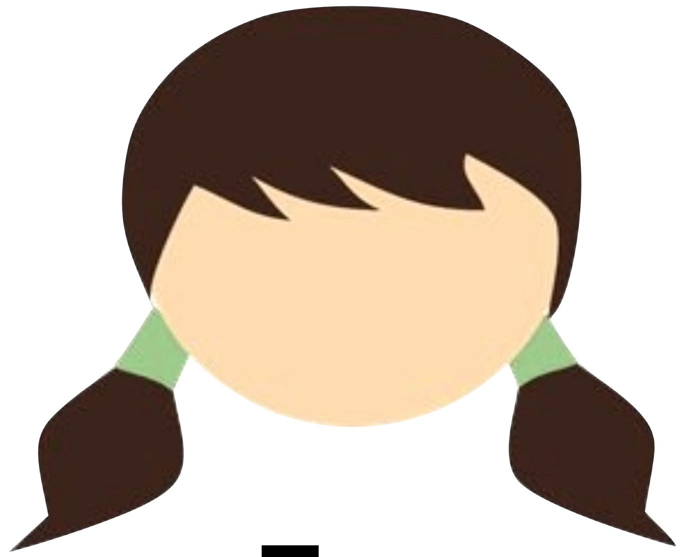
Security?



ct

Alice

Bob



Eve

Alice:

$m \in \{0,1\}^n$

$s \leftarrow\$ \{0,1\}^\lambda$

$k \leftarrow G(s)$

$ct \leftarrow m \oplus k$

Bob:

$s \leftarrow\$ \{0,1\}^\lambda$

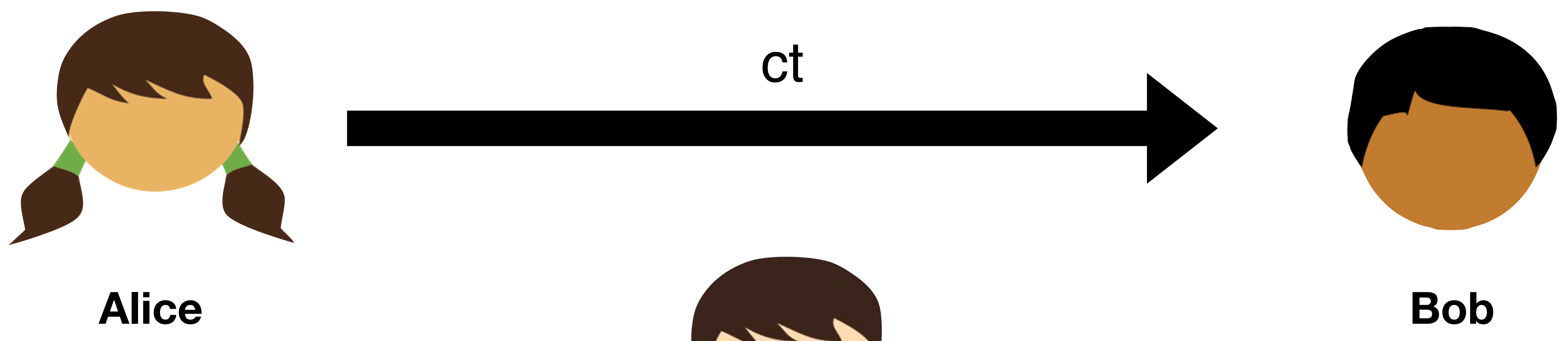
$k \leftarrow G(s)$

$m \leftarrow ct \oplus k$

Perfect Secrecy:

For every pair of messages $m_0, m_1 \in \mathrm{M}$ and every cipher text $c \in \mathrm{C}$:

$$\Pr_{k \leftarrow \mathrm{K}} [\ Enc(k, m_0) = c\ ] = \Pr_{k \leftarrow \mathrm{K}} [\ Enc(k, m_1) = c\ ]$$



$m \in \{0,1\}^n$

$s \leftarrow\$ \{0,1\}^\lambda$

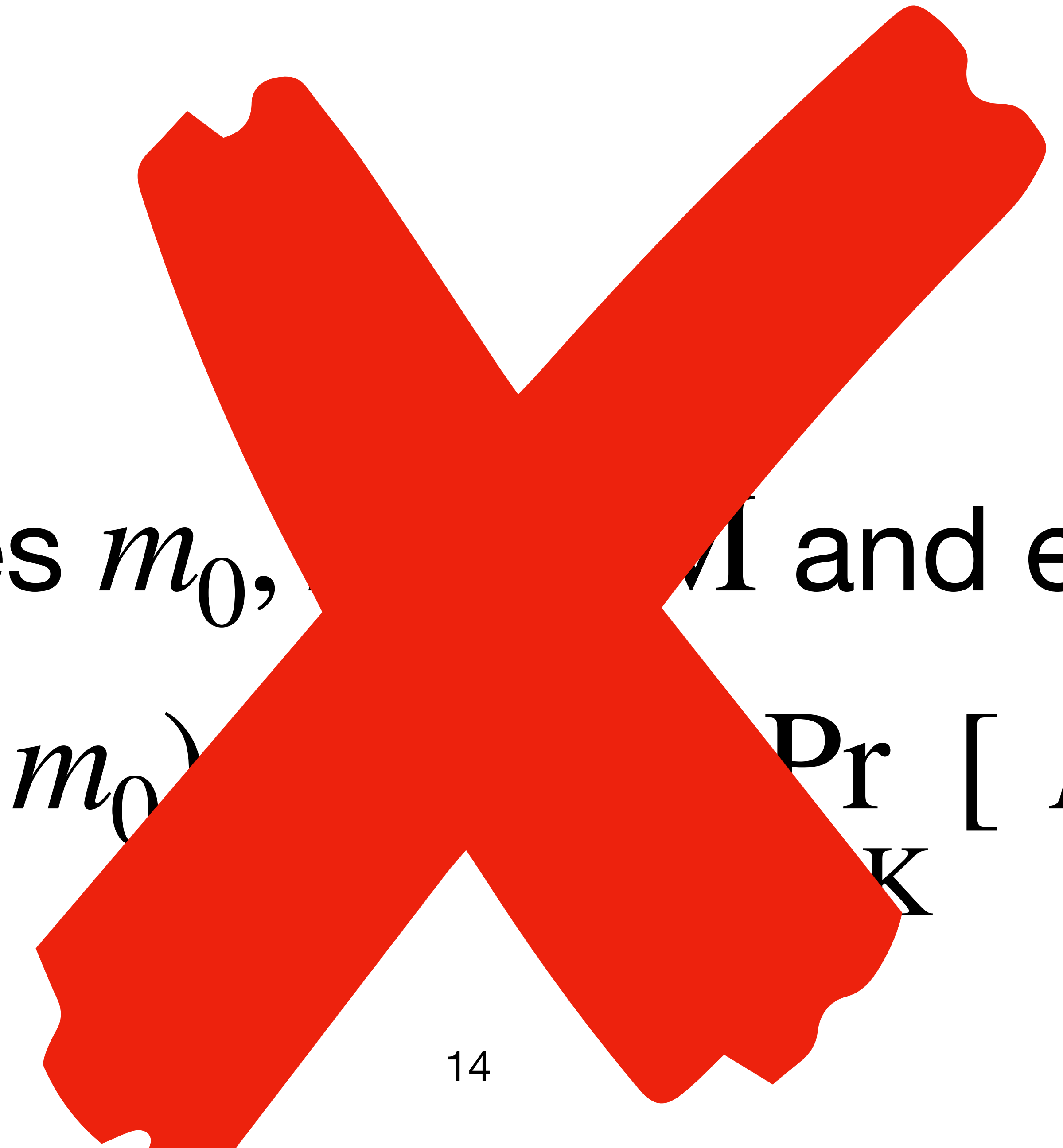
$k \leftarrow G(s)$

$ct \leftarrow m \oplus k$

$s \leftarrow\$ \{0,1\}^\lambda$

$k \leftarrow G(s)$

$m \leftarrow ct \oplus k$



Perfect Secrecy:

For every pair of messages $m_0, m_1 \in M$ and every cipher text $c \in C$:

$$\Pr_{k \leftarrow K}[\ Enc(k, m_0) = c\ ] = \Pr_{k \leftarrow K}[\ Enc(k, m_1) = c\ ]$$

A cipher (Enc, Dec) has **one-time semantic security** if:

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$$\overset{c}{\approx}$$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

Indistinguishability

$$L \overset{c}{\approx} R$$

Two programs L and R are **indistinguishable** if for *any* polynomial-time program A outputting a bit, the following quantity is negligible (in λ):

$$|\Pr[\ A \diamond L = 1\ ] - \Pr[\ A \diamond R = 1\ ]|$$

Interchangeability / Perfect Indistinguishability / Identically Distributed

$$L \equiv R$$

Two programs L and R are **interchangeable** if for *any* ~~polynomial-time~~ program A outputting a bit, the following holds:

$$\Pr[\ A \diamond L = 1\ ] = \Pr[\ A \diamond R = 1\ ]$$

Interchangeability

```
OTP(m0, m1):
  k ←$ {0,1}^n
  ct ← k ⊕ m0
  return ct
```

$\equiv$

```
OTP(m0, m1):
  k ←$ {0,1}^n
  ct ← k ⊕ m1
  return ct
```

Interchangeability

```
OTP(m0, m1):
  k ←$ {0,1}^n
  ct ← k ⊕ m0
  return ct
```

$\equiv$

```
OTP(m0, m1):
  k ←$ {0,1}^n
  ct ← k ⊕ m1
  return ct
```

Why?

one-time semantic security:

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$\overset{c}{\approx}$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

Perfect secrecy

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$\equiv$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$\overset{c}{\approx}$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

```
Enc(k, m):
  return m ⊕ G(k)
```

```
Dec(k, ct):
  return ct ⊕ G(k)
```

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$\overset{c}{\approx}$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

```
Enc(k, m):
  return m ⊕ G(k)
```

```
Dec(k, ct):
  return ct ⊕ G(k)
```

Goal: Prove if G is a PRG, then Enc/Dec satisfies one-time semantic security

```
eavesdrop(m0, m1):
  k ←$ {0,1}^λ
  r ← G(k)
  ct ← m0 ⊕ r
  return ct
```

$\overset{c}{\approx}$ **PRG Security**

```
eavesdrop(m0, m1):

  r ←$ {0,1}^n
  ct ← m0 ⊕ r
  return ct
```

$\equiv$ **Perfect Secrecy of one-time pad**

```
eavesdrop(m0, m1):

  r ←$ {0,1}^n
  ct ← m1 ⊕ r
  return ct
```

$\overset{c}{\approx}$ **PRG Security**

```
eavesdrop(m0, m1):
  k ←$ {0,1}^λ
  r ← G(k)
  ct ← m1 ⊕ r
  return ct
```

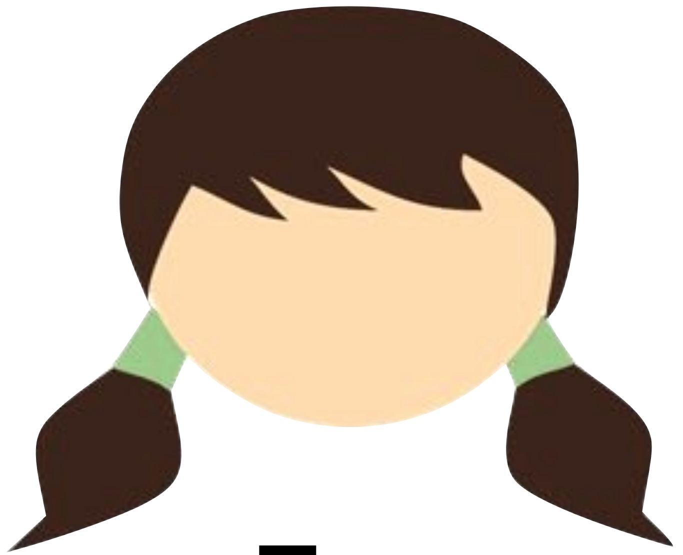
ct →

Alice

m ∈ $\{0,1\}^n$

k ←\$ $\mathscr{K}$

ct ← Enc(k, m)



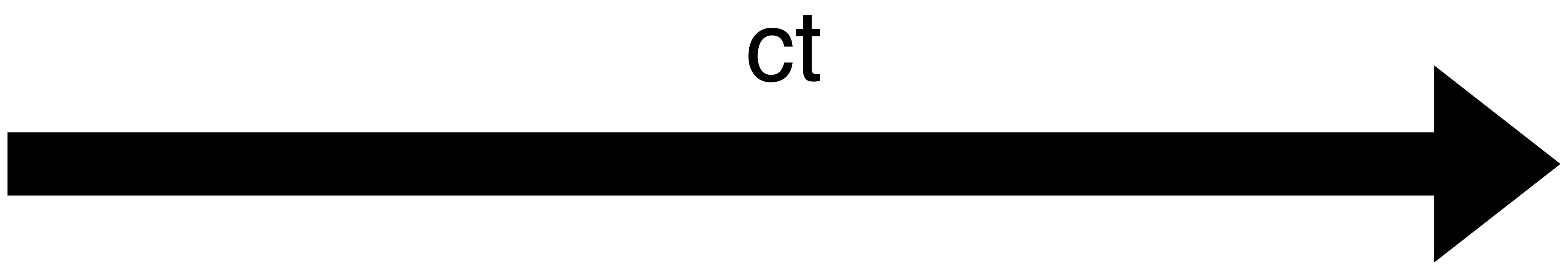
Eve

Bob

k ←\$ $\mathscr{K}$

m ← Dec(k, ct)

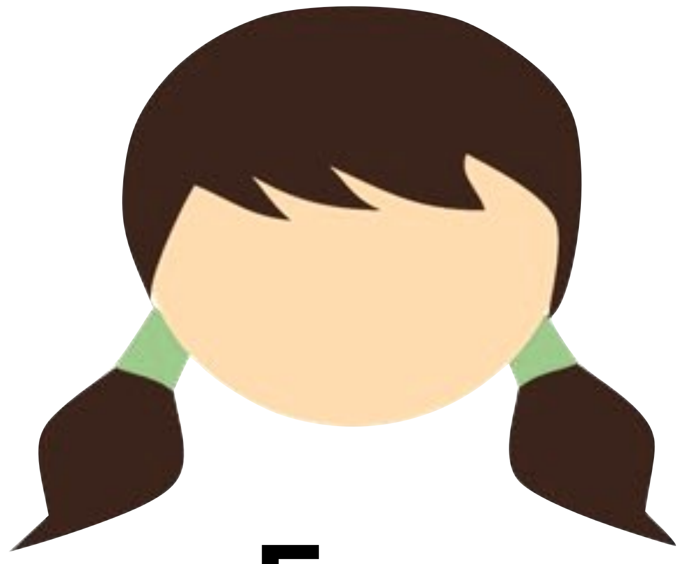
Alice



Bob

$\mathtt{m} \in \{0,1\}^n$

```
k ←$ 𝒦
ct ← Enc(k, m)
```



Eve

```
k ←$ 𝒦
m ← Dec(k, ct)
```

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m0)
  return ct
```

$\overset{c}{\approx}$

```
eavesdrop(m0, m1):
  k ←$ 𝒦
  ct ← Enc(k, m1)
  return ct
```

How does the security definition relate to our use-case?

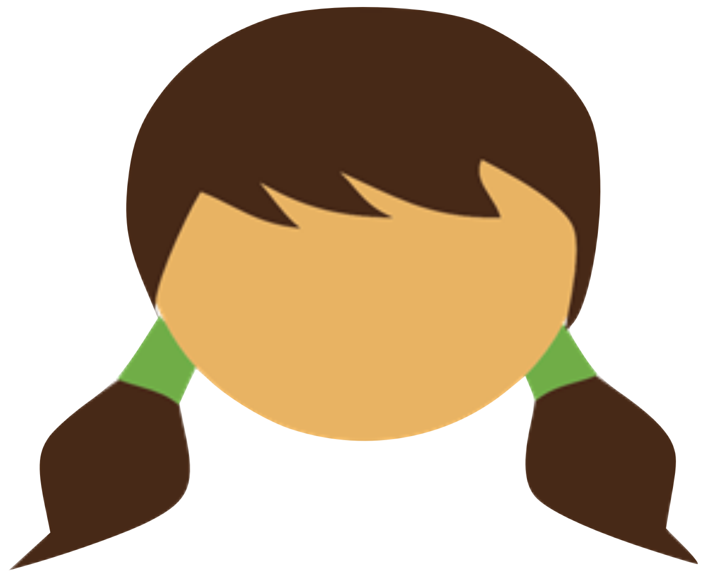
Now, Alice can send one long message to Bob, using only a short key

Now, Alice can send one long message to Bob, using only a short key

From here…
  More than one message?
  Authenticity?

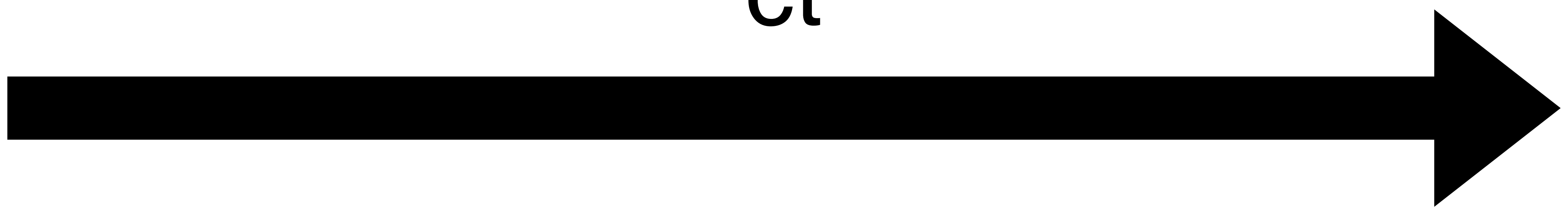
We will need new tools to get these

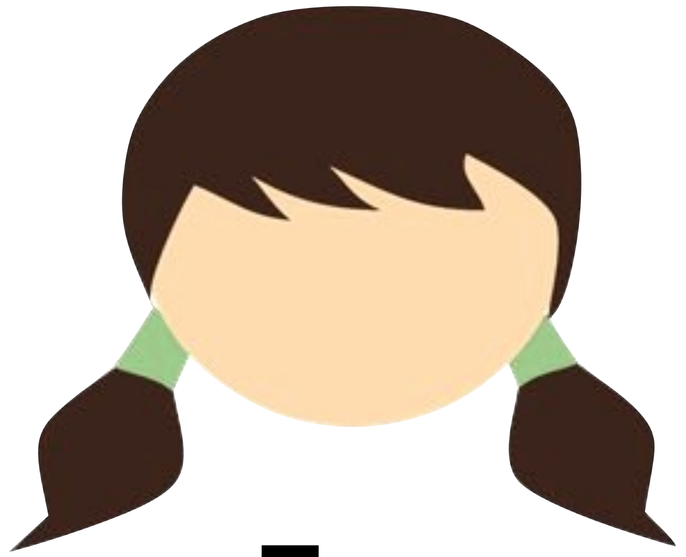
Pseudorandom Functions



Alice

ct





Eve

Bob

$\mathtt{m} \in \{0,1\}^n$

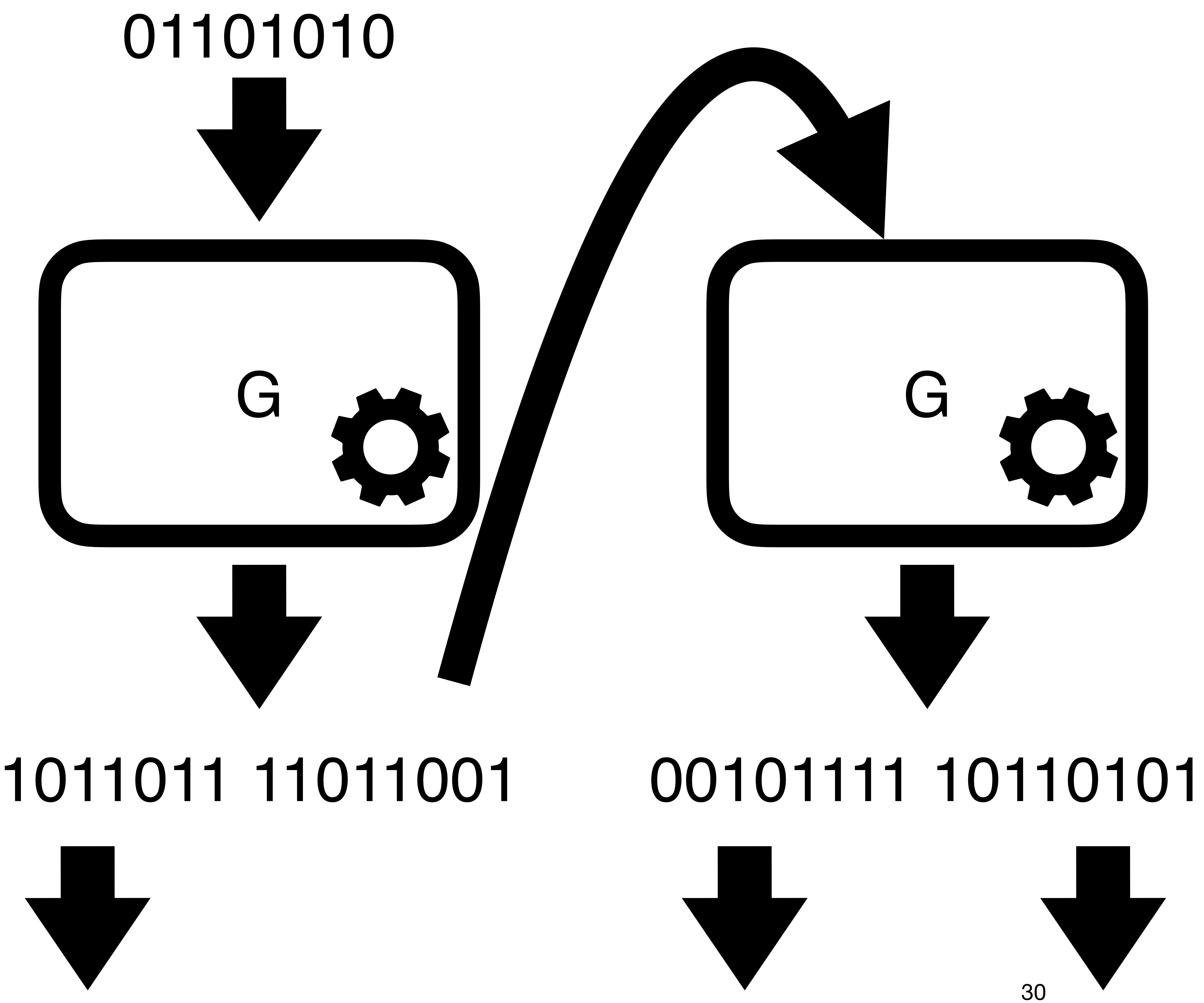
$\mathtt{k} \leftarrow\$ \{0,1\}^n$

$\mathtt{ct} \leftarrow \mathtt{m} \oplus \mathtt{k}$

$\mathtt{k} \leftarrow\$ \{0,1\}^n$

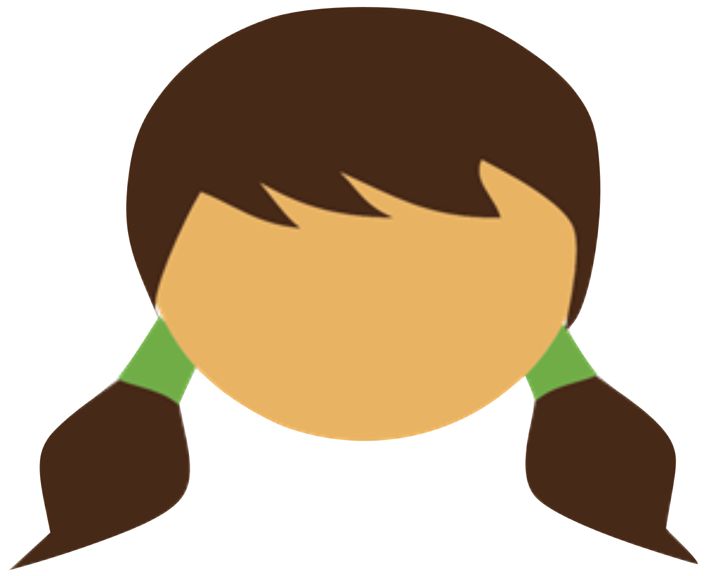
$\mathtt{m} \leftarrow \mathtt{ct} \oplus \mathtt{k}$

Stretching the output of a PRG



Call multiple times to get more randomness

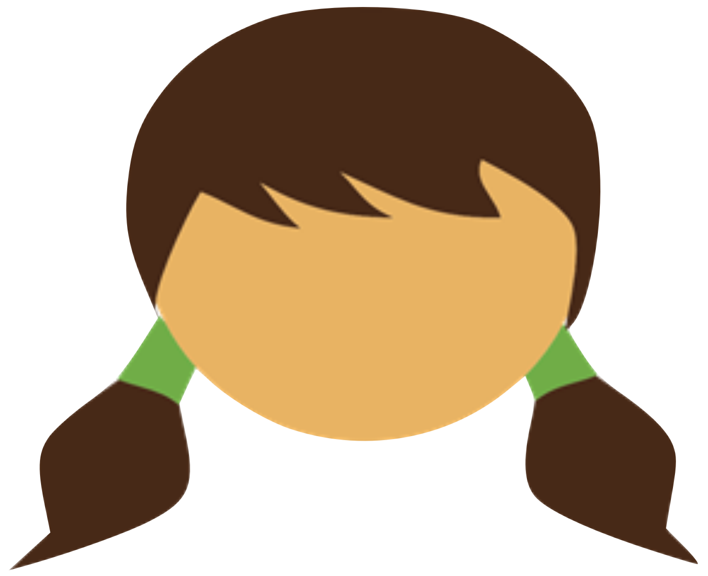
What about a cryptographic primitive that generates a lot of randomness "all at once"



Alice

Bob

0	01101000
1	11110000
2	10001110
3	01010100
4	11011010
…	…

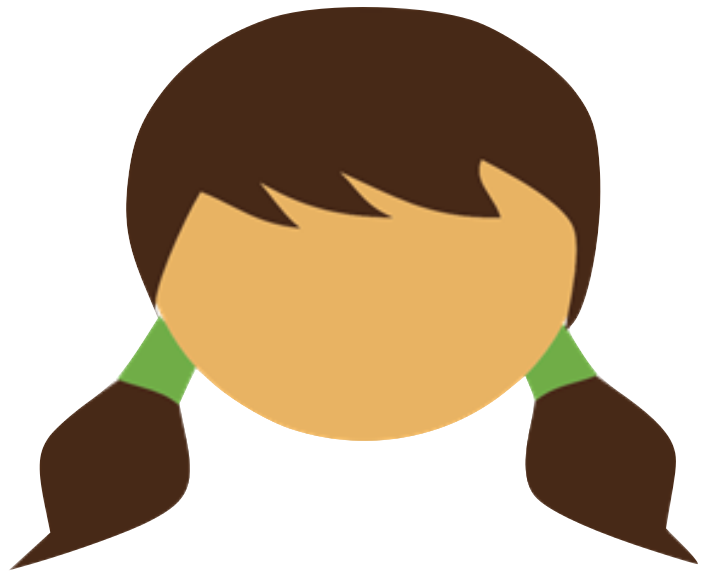


Alice

Bob

0	01101000
1	11110000
2	10001110
3	01010100
4	11011010
…	…

2^{λ} **rows**



Alice

Bob

0	01101000
1	11110000
2	10001110
3	01010100
4	11011010
…	…

A pseudorandom function (PRF) allows Alice and Bob to share a huge pseudorandom table via a short key

$$F : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^m$$

F is called a **pseudorandom function family** if the following indistinguishability holds:

```
k ←$ {0,1}^λ

apply(x):
  return F(k, x)
```

$$\overset{c}{\approx}$$

```
D ← empty-dictionary

apply(x):
  if x is not in D:
    D[x] ←$ {0,1}^m
  return D[x]
```

$$F : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^m$$

F is called a **pseudorandom function family** if the following indistinguishability holds:

```
k ←$ {0,1}^λ

apply(x):
  return F(k, x)
```

$\overset{c}{\approx}$

```
D ← empty-dictionary

apply(x):
  if x is not in D:
    D[x] ←$ {0,1}^m
  return D[x]
```

"Randomly sampling k emulates a huge random table"

$$F : \{0,1\}^\lambda \times \{0,1\}^n \to \{0,1\}^m$$

F is called a **pseudorandom function family** if the following indistinguishability holds:

```
k ←$ {0,1}^λ

apply(x):
  return F(k, x)
```

$$\overset{c}{\approx}$$

```
D ← empty-dictionary

apply(x):
  if x is not in D:
    D[x] ←$ {0,1}^m
  return D[x]
```

Closer to how real-world primitives are defined: We'll look at a candidate PRF ("The AES Block Cipher") next time

Given a PRF, build a PRG

PRG → PRF

PRG ← PRF

Given a PRG, build a PRF

Given a PRF, build a PRG

“Straightforward”, homework problem

PRG → PRF

PRG ← PRF

Given a PRG, build a PRF

Goldreich-Goldwasser-Micali construction

$$f : \{0,1\}^n \rightarrow \{0,1\}^m$$

f is called a **one-way function** if for any PPT program A and for all inputs x the following probability is negligible (in n):

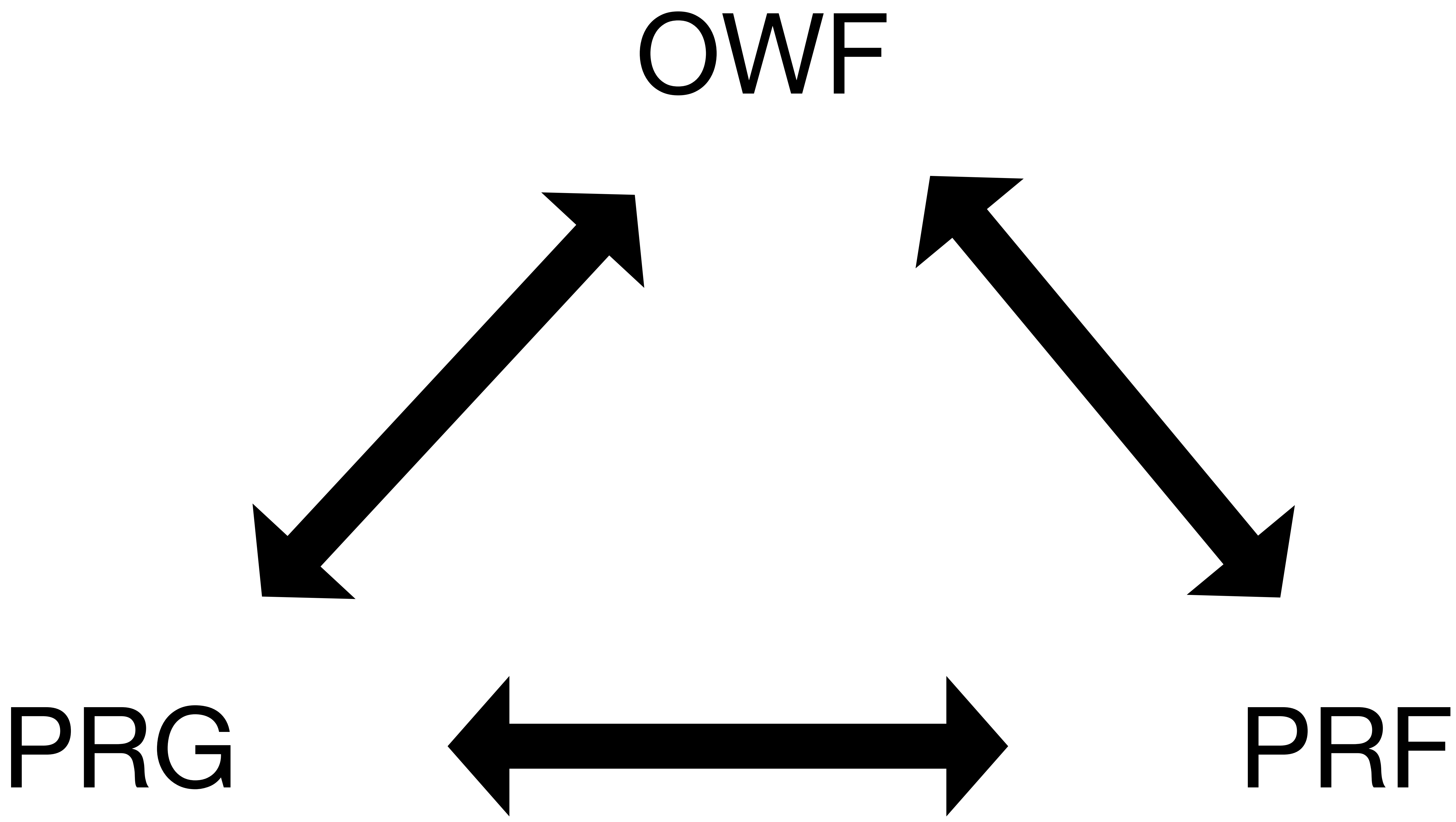
$$\Pr_{x \leftarrow \{0,1\}^n} \left[f(A(f(x)) = f(x) \right]$$

$$f : \{0,1\}^n \rightarrow \{0,1\}^m$$

f is called a **one-way function** if for any PPT program A and for all inputs x the following probability is negligible (in n):

$$\Pr_{x \leftarrow \{0,1\}^n} \left[f(A(f(x)) = f(x) \right]$$

“f is hard to invert”



OWFs exist $\implies P \neq NP$

Today's objectives

See an attack on a "PRG"

Use PRG to define a new cipher

Define interchangeability

Define one-time semantic security

Prove our cipher satisfies one-time semantic security

Introduce Pseudorandom Functions (PRFs)