COS433/Math 473: Cryptography

Mark Zhandry

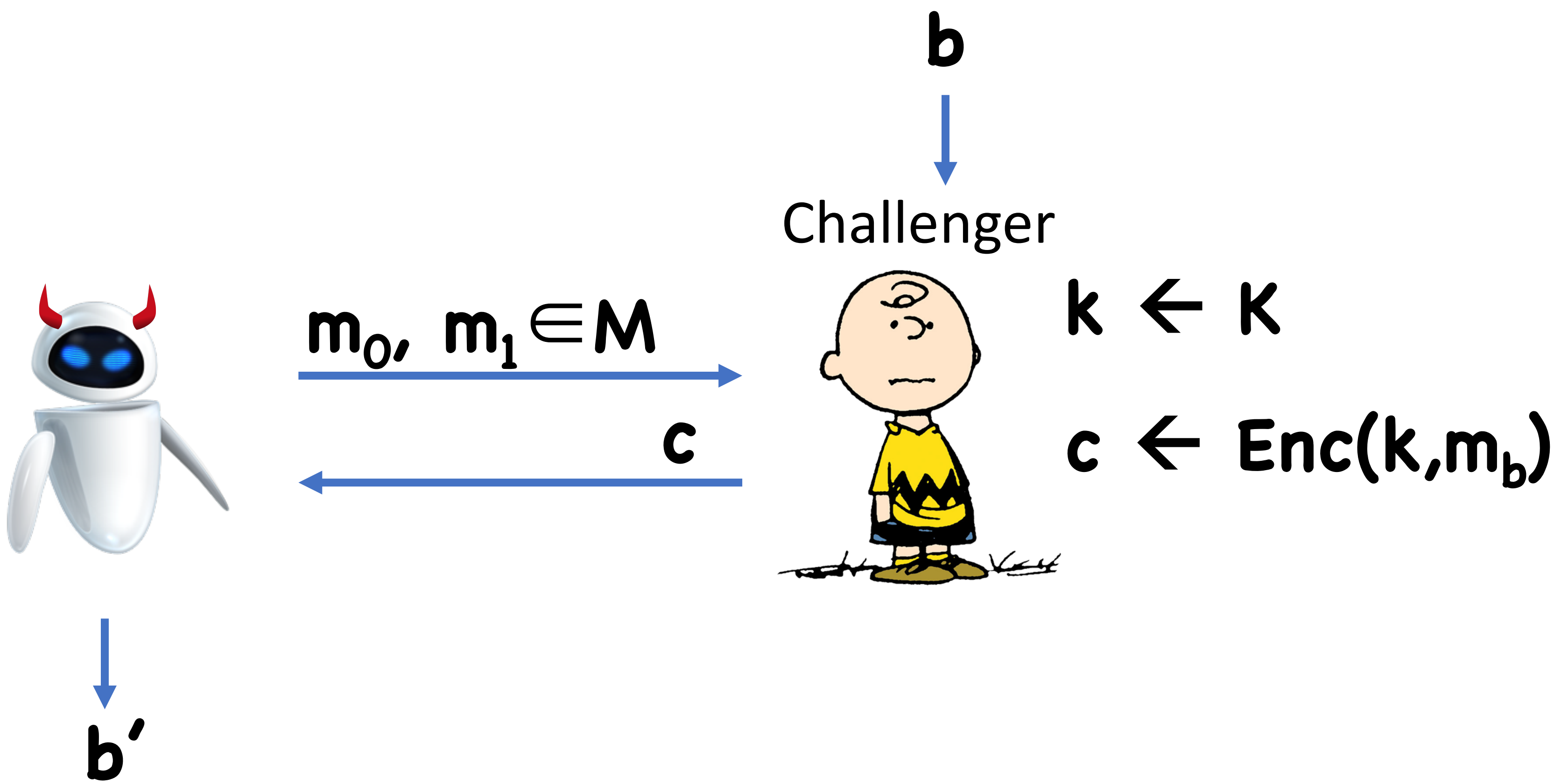
Princeton University

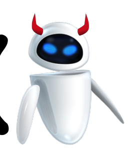
Spring 2017

Previously on COS 433…

Security Experiment/Game

(One-time setting)



IND-Exp$_b$(**)**

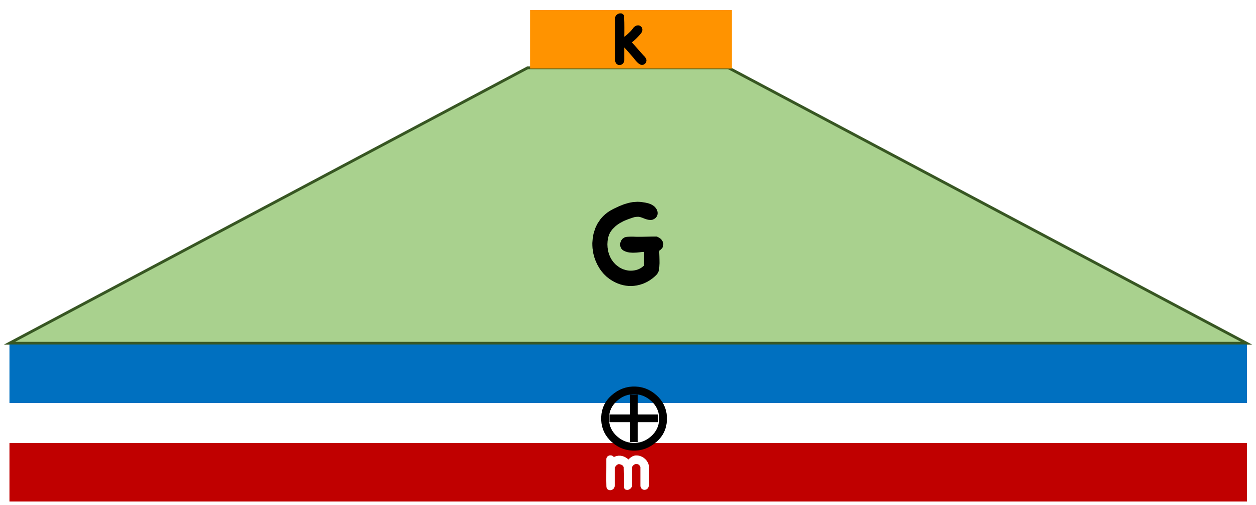
Security Definition (One-time setting)

Definition: (Enc, Dec) has **(t,ε)-ciphertext indistinguishability** if, for all  running in time at most **t**

$$\left|\ \Pr[1 \leftarrow \text{IND-Exp}_0(\ \cdot\ )\ ] - \Pr[1 \leftarrow \text{IND-Exp}_1(\ \cdot\ )\ ]\ \right| \le \varepsilon$$

Construction with |k| << |m|

Idea: use OTP, but have key generated by some expanding function **G**



What Do We Want Out of **G**?

Definition: G:$\{0,1\}^\lambda \rightarrow \{0,1\}^n$ is a **(t,ε)-secure pseudorandom generator** (PRG) if:

- **$n > \lambda$**
- **G** is deterministic
- For all  running in time at most **t**,

| Pr[ **$(G(s))=1 : s \leftarrow \{0,1\}^\lambda$]**

$-$ Pr[ **$(x)=1 : x \leftarrow \{0,1\}^n$] |** $\leq \varepsilon$

Reminder: Kerckhoffs's Principle

Kerckhoffs's Principle: A cryptosystem should be secure even if everything about the system, except the key, is public knowledge.

Applies to any crypto object we'll see in this course

For PRGs, the "key" is just the input to the function

Secure PRG → Ciphertext Indistinguishability

$K = \{0,1\}^\lambda$
$M = \{0,1\}^n$
$C = \{0,1\}^n$

$\mathrm{Enc}(k,m) = \mathrm{PRG}(k) \oplus m$
$\mathrm{Dec}(k,c) = \mathrm{PRG}(k) \oplus c$

Security?

Intuitively, security is obvious:

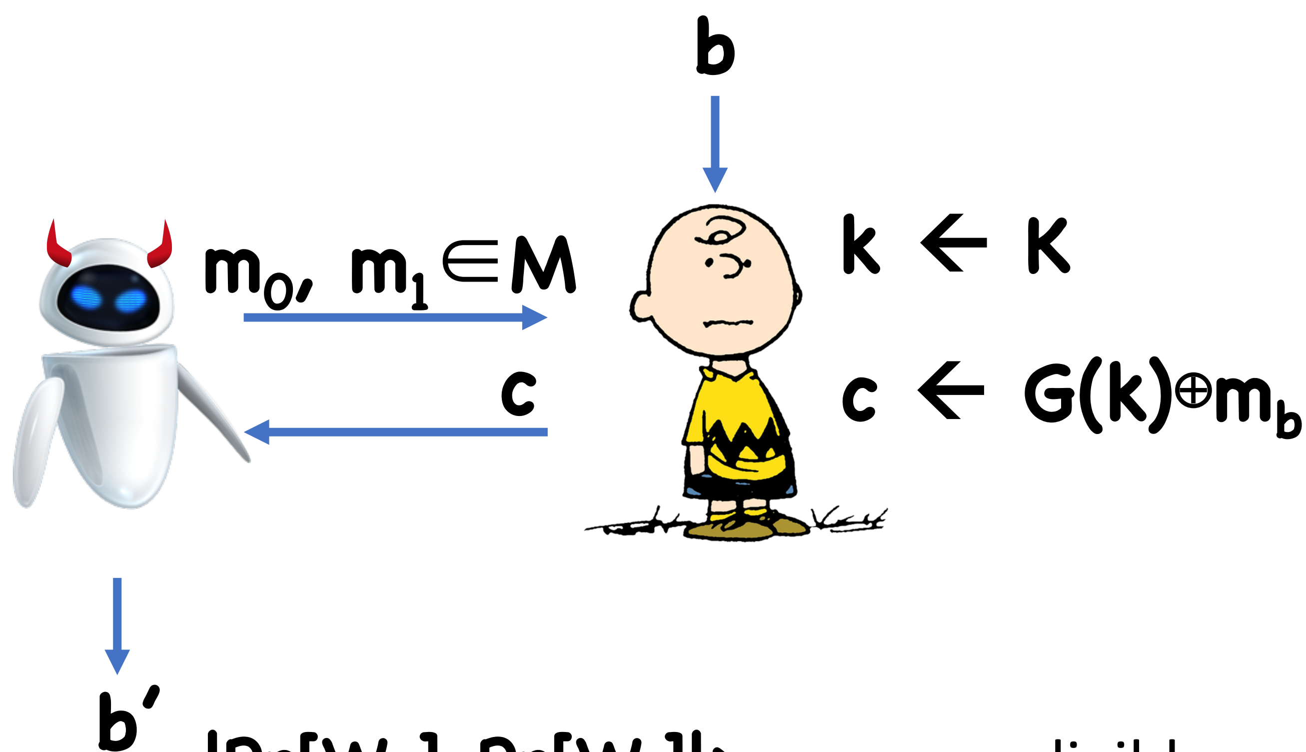
- **PRG(k)** ”looks” random, so should completely hide **m**
- However, formalizing this argument is non-trivial.

Solution: reductions

- Assume toward contradiction an adversary for the encryption scheme, derive an adversary for the PRG

Security

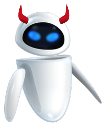
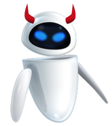
Assume towards contradiction that there is a  such that

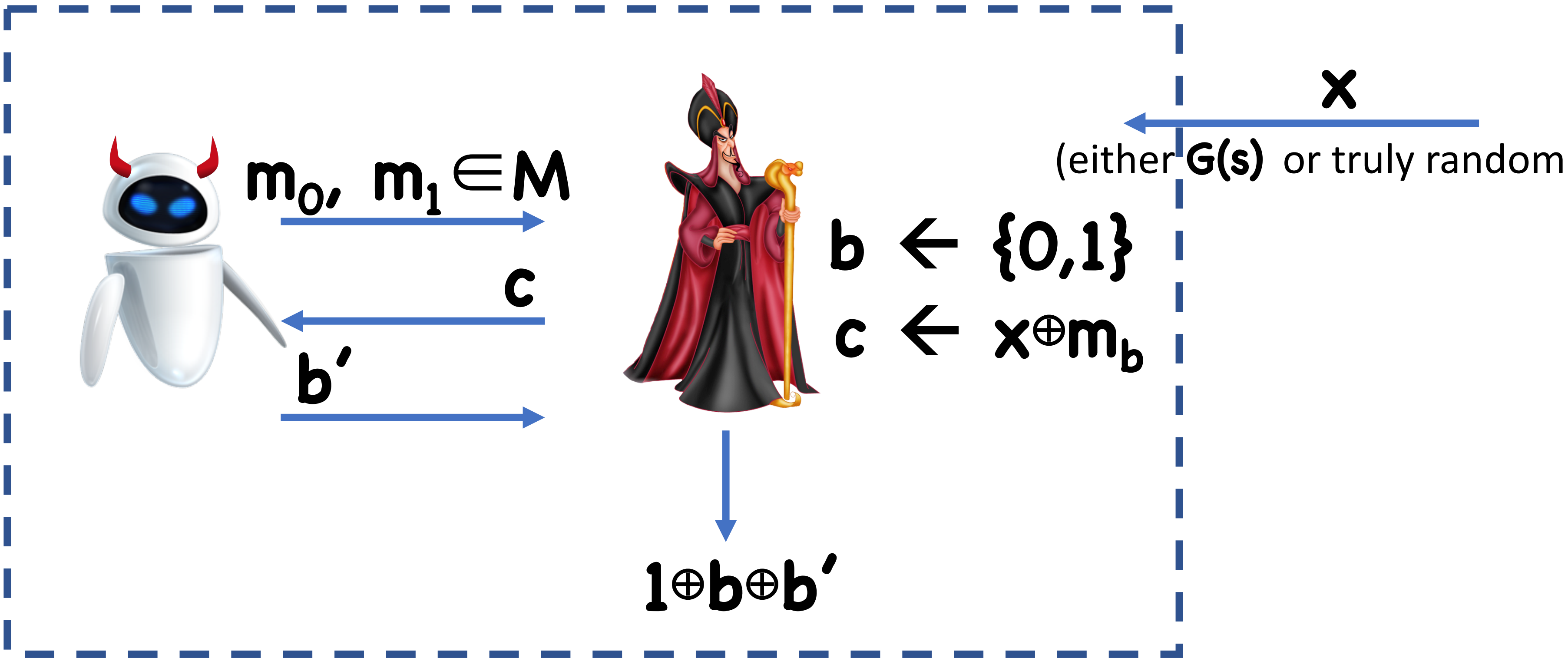


$|\Pr[W_0] - \Pr[W_1]| \geq \varepsilon$, non-negligible

W_b: $b' = 1$ in IND-Exp_b

Security

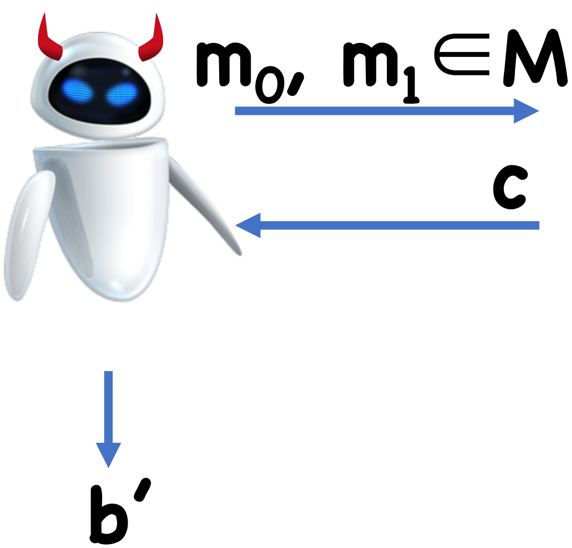
Use  to build .  will run  as a subroutine, and pretend to be 



Security

Case 1: $\mathbf{x = PRG(s)}$ for a random seed $\mathbf{s}$

-  “sees” $\mathbf{IND\text{-}Exp_b}$ for a random bit $\mathbf{b}$



$\mathbf{b \leftarrow \{0,1\}}$

$\mathbf{s \leftarrow K}$

$\mathbf{c \leftarrow PRG(s) \oplus m_b}$

Security

Case 1: $\mathbf{x} = \mathbf{PRG(s)}$ for a random seed $\mathbf{s}$

-  “sees” $\mathbf{IND\text{-}Exp_b}$ for a random bit $\mathbf{b}$
- $\mathbf{Pr[1 \oplus b \oplus b' = 1] = Pr[b = b']}$

$$\begin{aligned} &= \tfrac{1}{2} \Pr[b'=1 \mid b=1] \\ &\qquad + \tfrac{1}{2} (1 - \Pr[b'=1 \mid b=0]) \\ &= \tfrac{1}{2}(1 + \Pr[W_0] - \Pr[W_1]) \\ &= \tfrac{1}{2}(1 \pm \varepsilon) \end{aligned}$$

Security

Case 2: $\mathbf{x}$ is truly random

-  "sees" OTP encryption

$m_0, m_1 \in M_\lambda$ →

← c

↓ b'

$b \leftarrow \{0,1\}$

$x \leftarrow \{0,1\}^n$

$c \leftarrow x \oplus m_b$

Security

Case 2: $\mathbf{x}$ is truly random

-  "sees" OTP encryption
- Therefore $\Pr[b'=1 \mid b=0] = \Pr[b'=1 \mid b=1]$
- $\Pr[1 \oplus b \oplus b'=1] = \Pr[b=b']$

$$= \tfrac{1}{2} \Pr[b'=1 \mid b=1] + \tfrac{1}{2}(1 - \Pr[b'=1 \mid b=0])$$

$$= \tfrac{1}{2}$$

Security

Putting it together:

- $\Pr[$  $(G(s))=1 : s \leftarrow \{0,1\}^\lambda] = \frac{1}{2}(1 \pm \varepsilon(\lambda))$
- $\Pr[$  $(x)=1 : x \leftarrow \{0,1\}^n] = \frac{1}{2}$
- Absolute Difference: $\frac{1}{2}\varepsilon$, $\Rightarrow$ Contradiction!

Security

Thm: If **G** is a **(t+t',ε/2)**-secure PRG, then **(Enc,Dec)** is has **(t,ε)**-ciphertext indistinguishability, where **t'** is the time to:

- Flip a random bit **b**
- XOR two **n**-bit strings

Security

Thm: If **G** is a **(t+poly,ε/2)**-secure PRG, then **(Enc,Dec)** is has **(t,ε)**-ciphertext indistinguishability

An Alternate Proof: Hybrids

Idea: define sequence of “hybrid” experiments “between” $\textbf{IND-Exp}_0$ and $\textbf{IND-Exp}_1$

In each hybrid, make small change from previous hybrid

Hopefully, each small change is undetectable

Using triangle inequality, overall change from $\textbf{IND-Exp}_0$ and $\textbf{IND-Exp}_1$ is undetectable

An Alternate Proof: Hybrids

Hybrid 0: IND-Exp$_0$

$m_0, m_1 \in M$

c

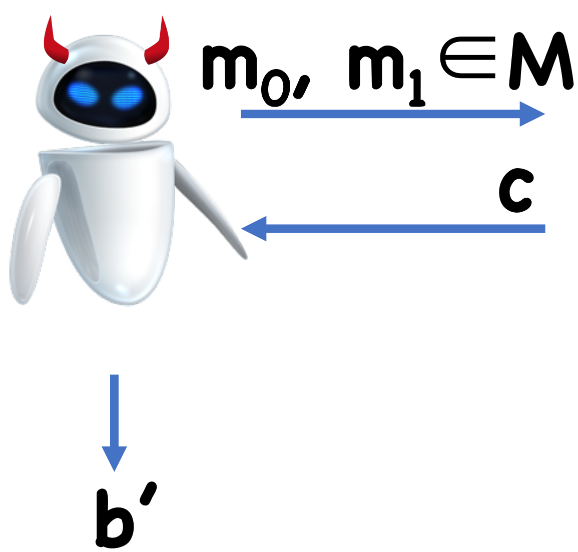
b'

$k \leftarrow K$

$c \leftarrow G(k) \oplus m_0$

An Alternate Proof: Hybrids

Hybrid 1:

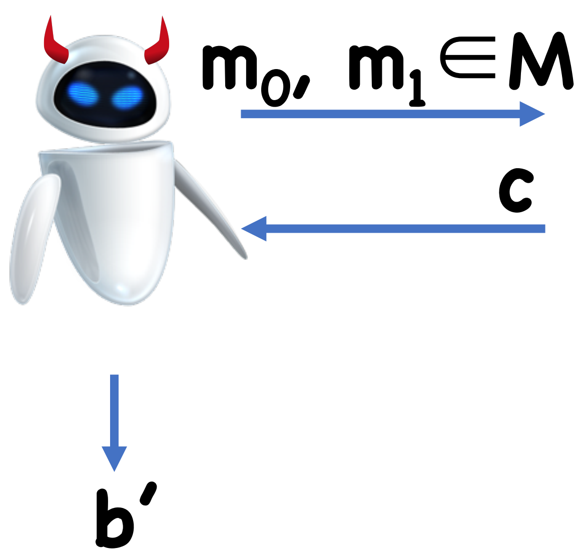


$x \leftarrow \{0,1\}^n$

$c \leftarrow x \oplus m_0$

An Alternate Proof: Hybrids

Hybrid 2:



$x \leftarrow \{0,1\}^n$

$c \leftarrow x \oplus m_1$

An Alternate Proof: Hybrids

Hybrid 3: IND-Exp$_1$

$m_0, m_1 \in M$ →

← c

↓ b'

$k \leftarrow K$

$c \leftarrow G(k) \oplus m_1$

An Alternate Proof: Hybrids

$| \Pr[b'=1 : \text{IND-Exp}_0] - \Pr[b'=1 : \text{IND-Exp}_1] |$

$= | \Pr[b'=1 : \text{Hyb } 0] - \Pr[b'=1 : \text{Hyb } 3] |$

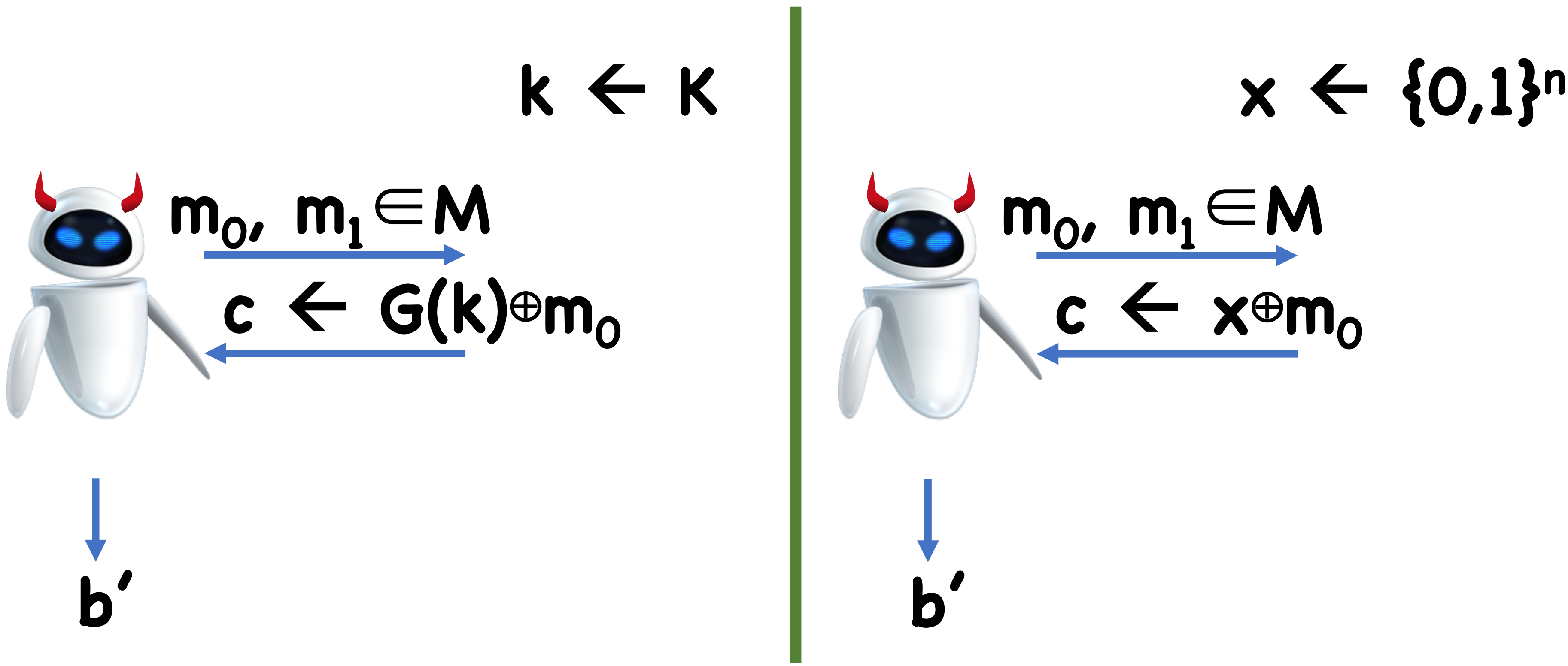
$\leq | \Pr[b'=1 : \text{Hyb } 0] - \Pr[b'=1 : \text{Hyb } 1] |$
$+ | \Pr[b'=1 : \text{Hyb } 1] - \Pr[b'=1 : \text{Hyb } 2] |$
$+ | \Pr[b'=1 : \text{Hyb } 2] - \Pr[b'=1 : \text{Hyb } 3] |$

If $|\Pr[b'=1:\text{IND-Exp}_0]-\Pr[b'=1:\text{IND-Exp}_1]| \geq \varepsilon$,
Then for some $i=0,1,2$,

$$|\Pr[b'=1:\text{Hyb } i]-\Pr[b'=1:\text{Hyb } i+1]| \geq \varepsilon/3$$

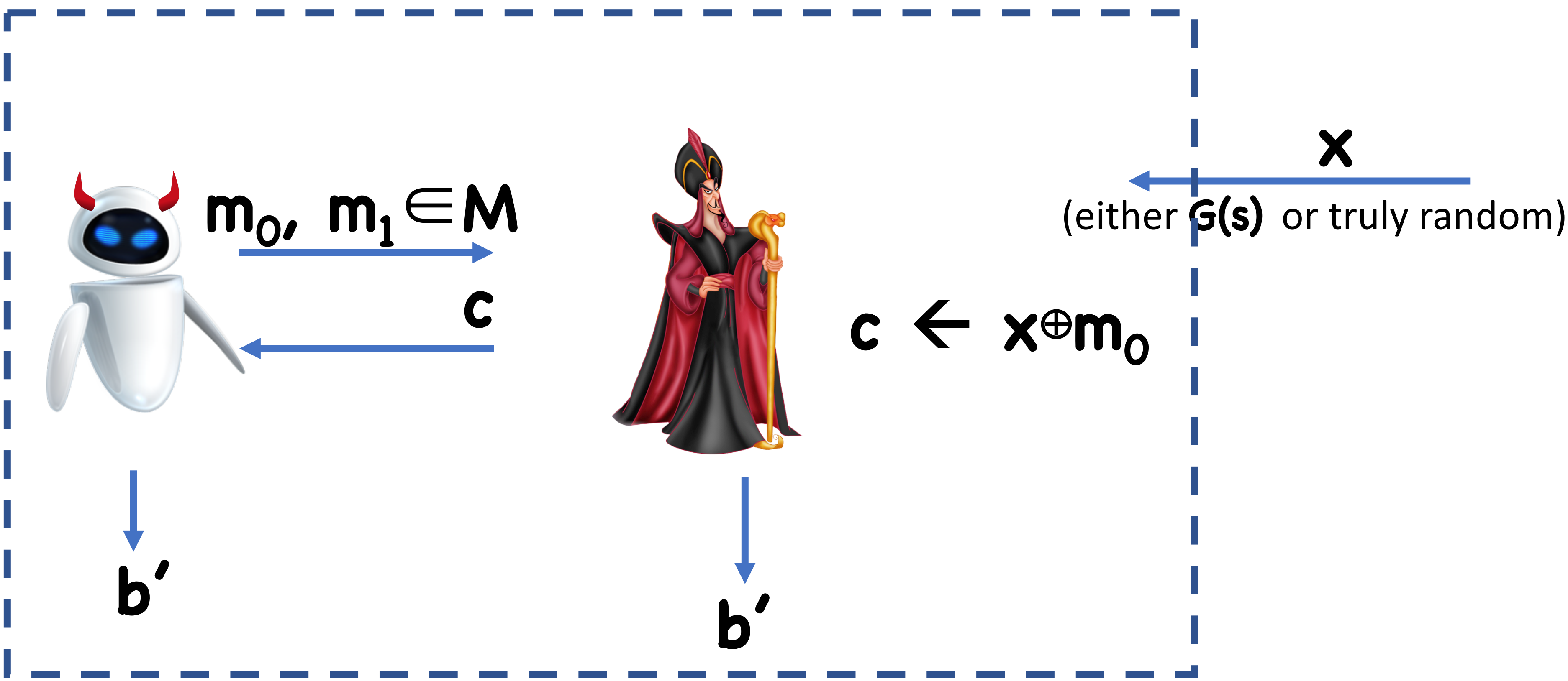
An Alternate Proof: Hybrids

Suppose  distinguishes **Hybrid 0** from **Hybrid 1** with advantage **ε/3**



An Alternate Proof: Hybrids

Suppose  distinguishes **Hybrid 0** from **Hybrid 1** with advantage **ε/3** ⇒ Construct 



An Alternate Proof: Hybrids

Suppose  distinguishes **Hybrid 0** from **Hybrid 1** with advantage **ε/3** ⇒ Construct 

If  is given **G(s)** for a random **s**,  sees **Hybrid 0**

If  is given x for a random **x**,  sees **Hybrid 1**

Therefore, advantage of  is equal to advantage of  which is at least **ε/3** ⇒ Contradiction!

An Alternate Proof: Hybrids

Suppose  distinguishes **Hybrid 1** from **Hybrid 2** with advantage **ε/3**

$x \leftarrow \{0,1\}^n$

$m_0, m_1 \in M \rightarrow$

$\leftarrow c \leftarrow x \oplus m_0$

$\downarrow b'$

$x \leftarrow \{0,1\}^n$

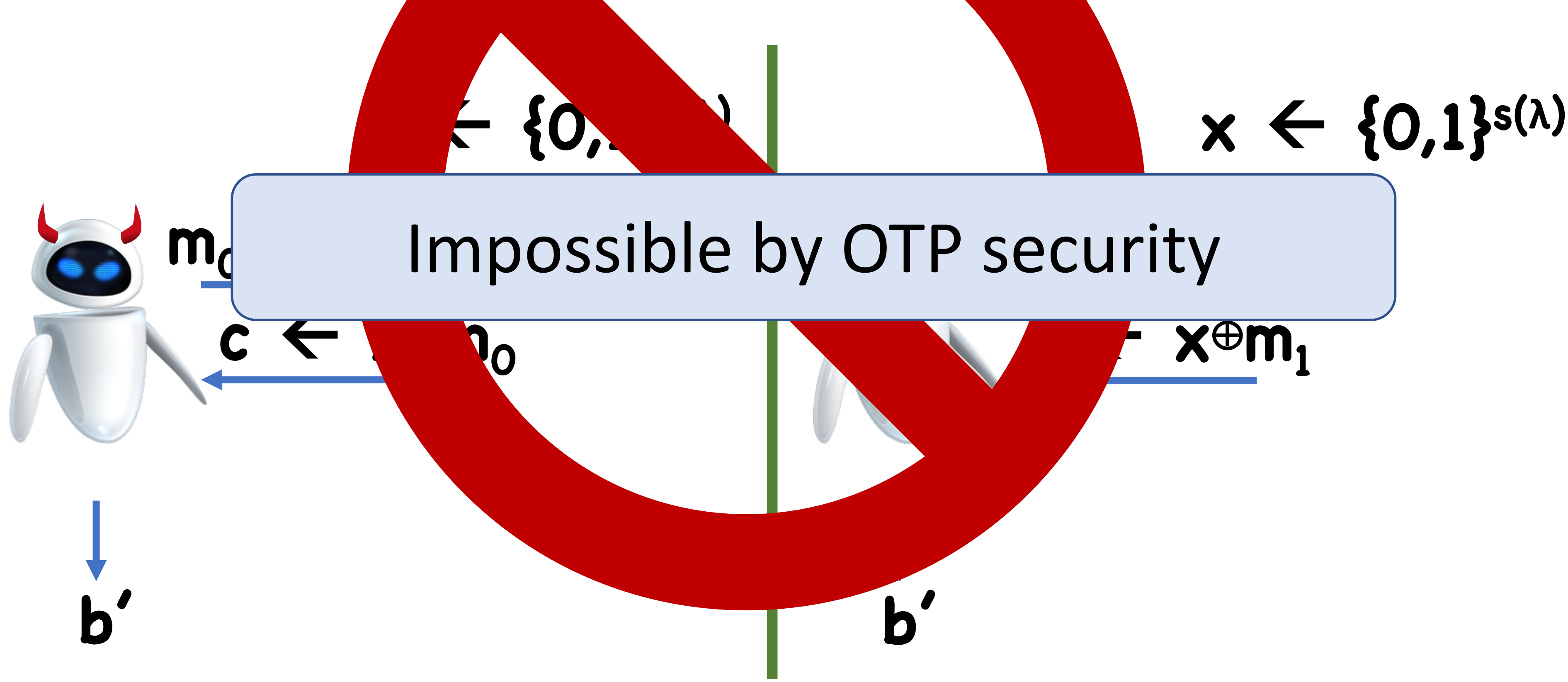
$m_0, m_1 \in M \rightarrow$

$\leftarrow c \leftarrow x \oplus m_1$

$\downarrow b'$

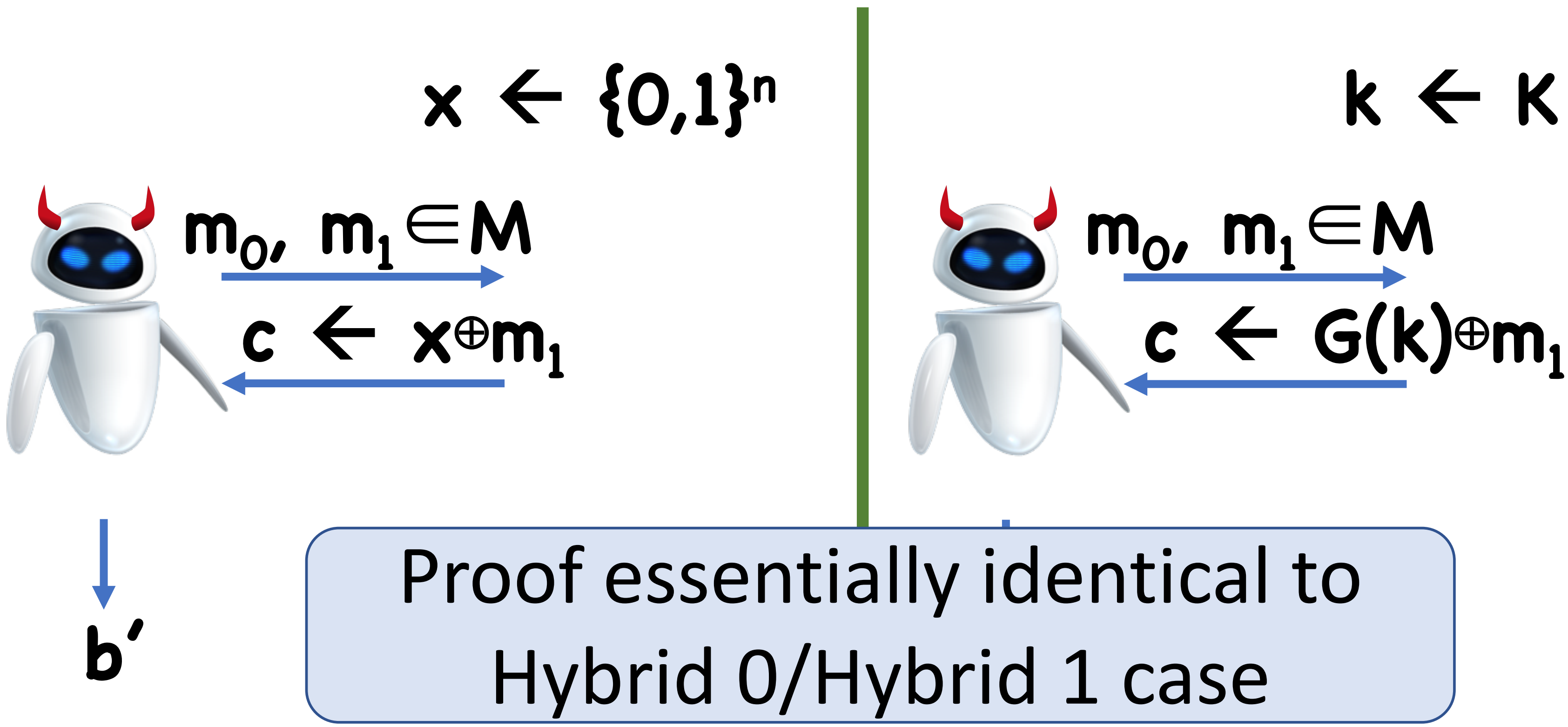
An Alternate Proof: Hybrids

Suppose disti… …1 from **Hybrid 2**
with advantag… …)/3**



An Alternate Proof: Hybrids

Suppose distinguishes **Hybrid 2** from **Hybrid 3** with advantage **ε/3**

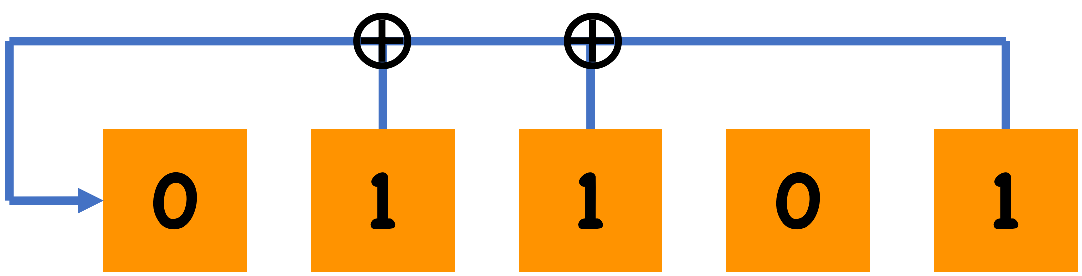


How do we build PRGs?

Linear Feedback Shift Registers

In each step,

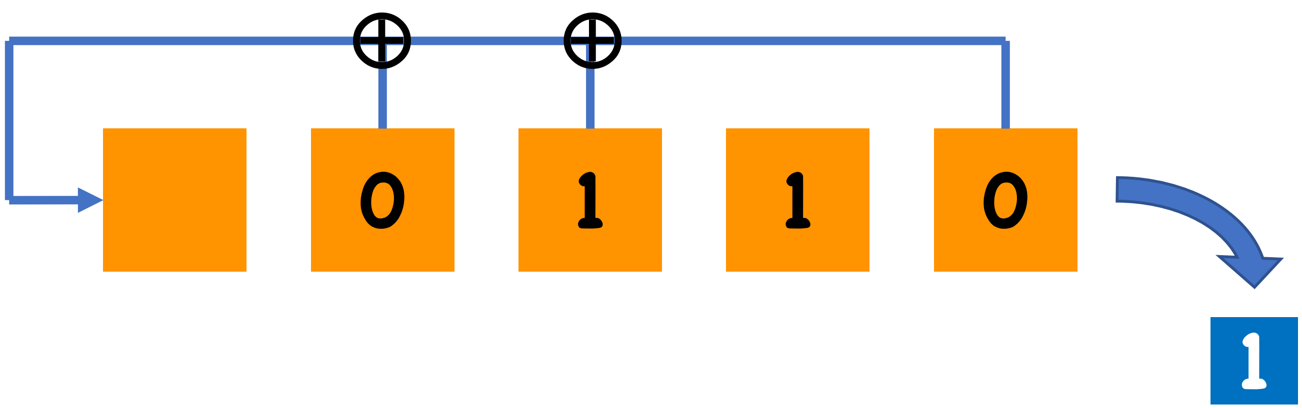
- Last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

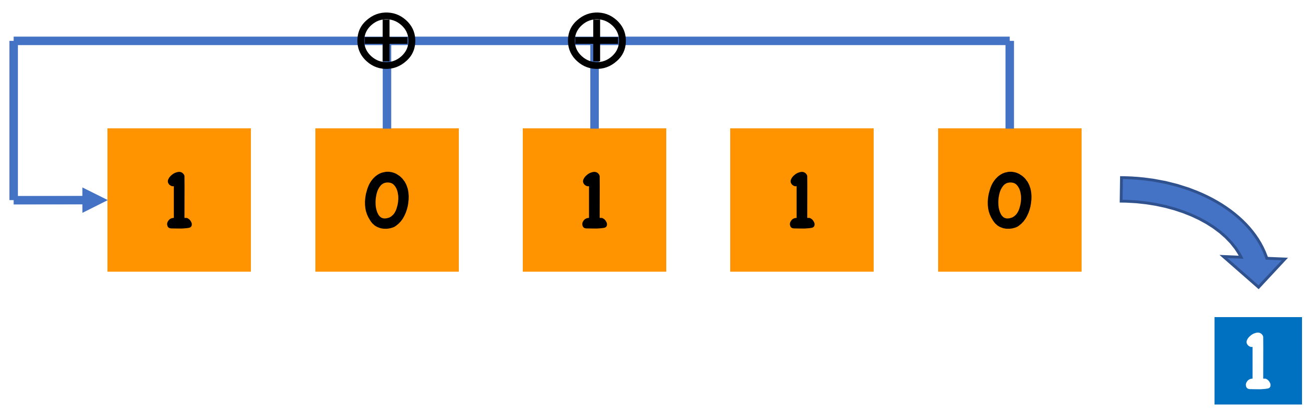
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

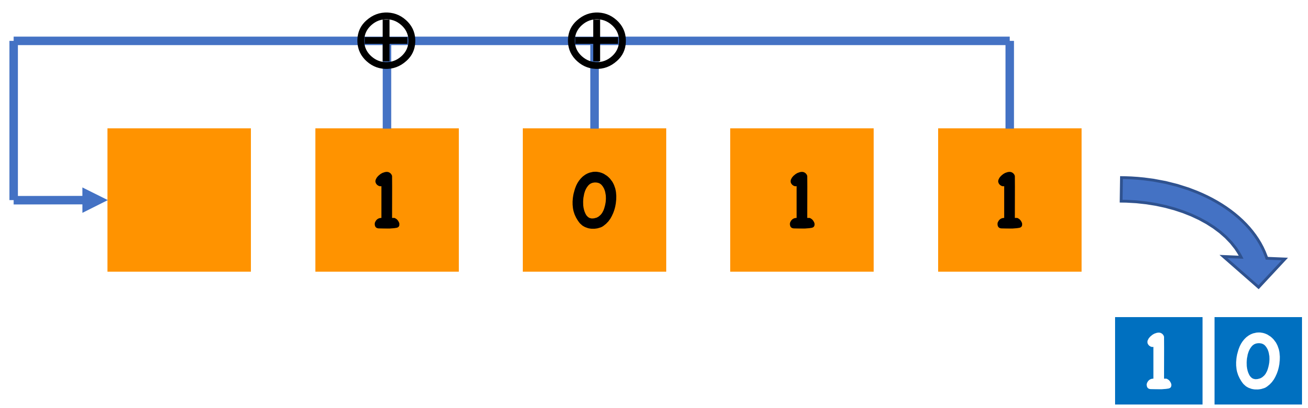
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

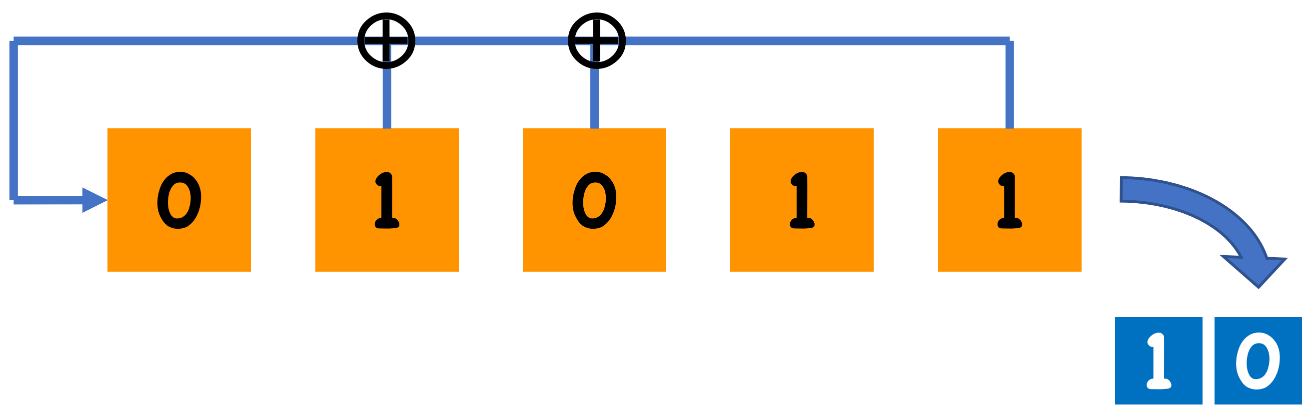
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

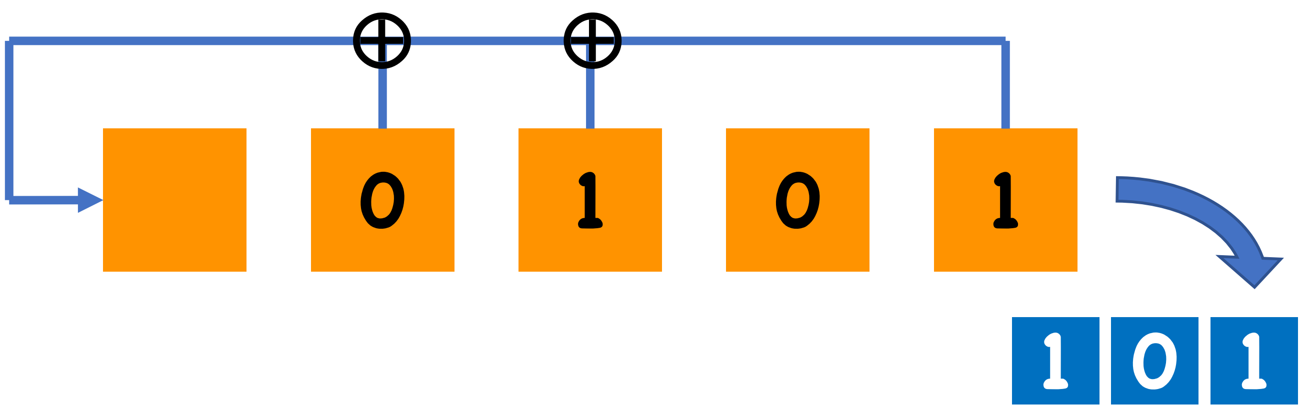
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

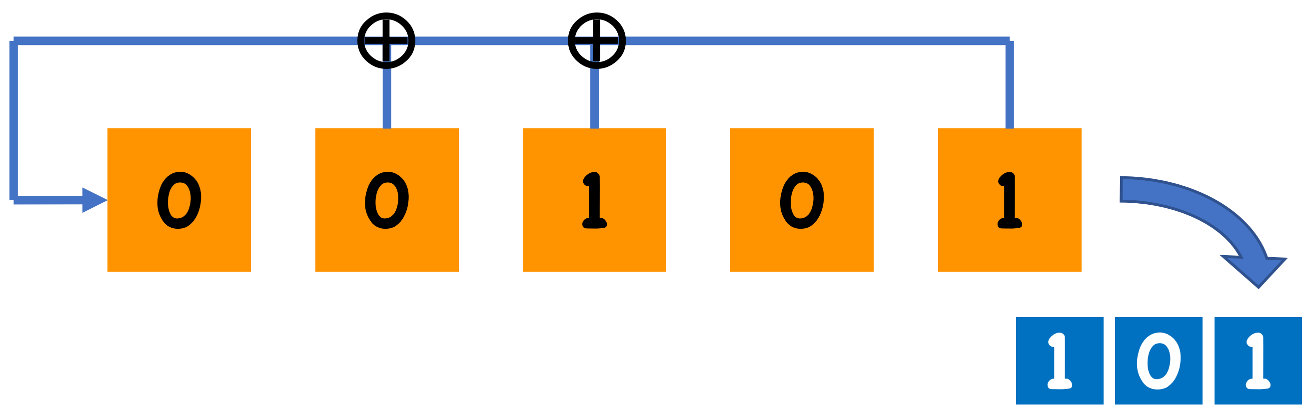
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

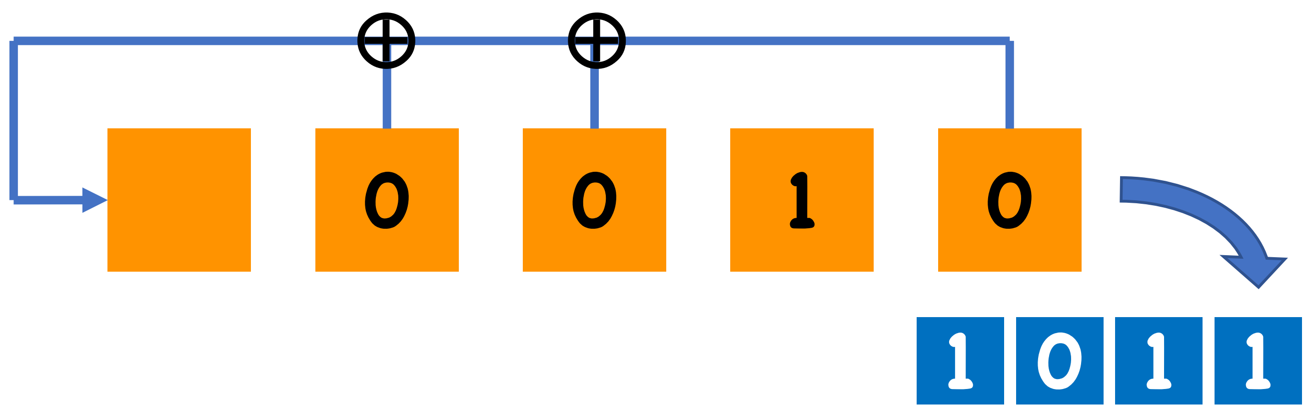
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

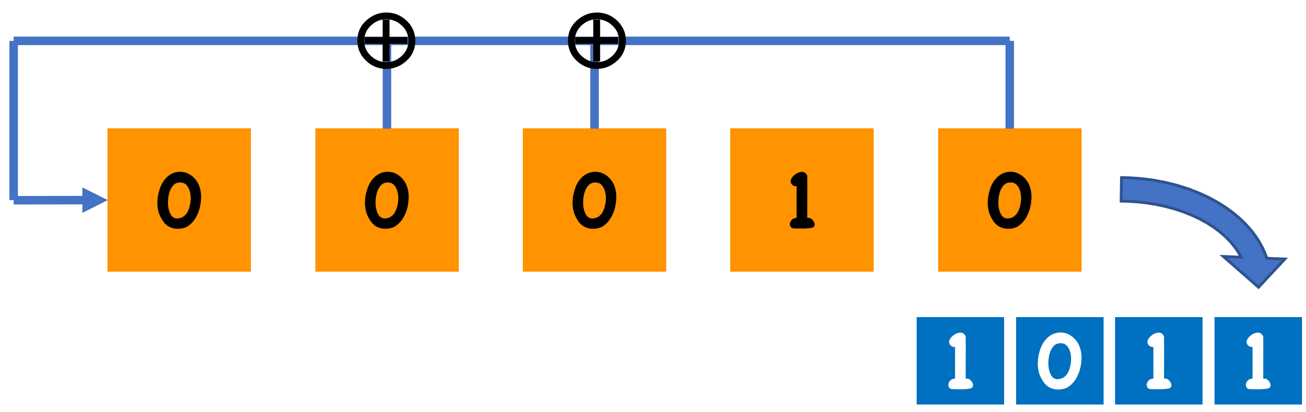
- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



Linear Feedback Shift Registers

In each step,

- last bit of state is removed and outputted
- Rest of bits are shifted right
- First bit is XOR of subset of remaining bits



…

Linear Feedback Shift Registers

Are LFSR's secure PRGs?

Linear Feedback Shift Registers

Are LFSR's secure PRGs?

No!

First **n** bits of output = initial state

$\mathbf{x}$ →

Write $\mathbf{x} = \mathbf{x}_1,\ldots,\mathbf{x}_n, \mathbf{x}'$

Initialize LFSB to have state $\mathbf{x}_1,\ldots,\mathbf{x}_n$

Run LFSB for $|\mathbf{x}|$ steps, obtaining $\mathbf{y}$

Check if $\mathbf{y} = \mathbf{x}$

PRGs should be Unpredictable

More generally, it should be hard, given some bits of output, to predict subsequent bits

Definition: G is **(t,p,ε)–unpredictable** if, for all  running in time at most **t**,

$\left| \Pr[G(s)_{p+1} \leftarrow$  $(G(s)_{[1,p]})] - \frac{1}{2} \right| \leq \varepsilon$

PRGs should be Unpredictable

More generally, it should be hard, given some bits of output, to predict subsequent bits

Theorem: G is **unpredictable** iff it is **pseudorandom**

Proof

Pseudorandomness → Unpredictability

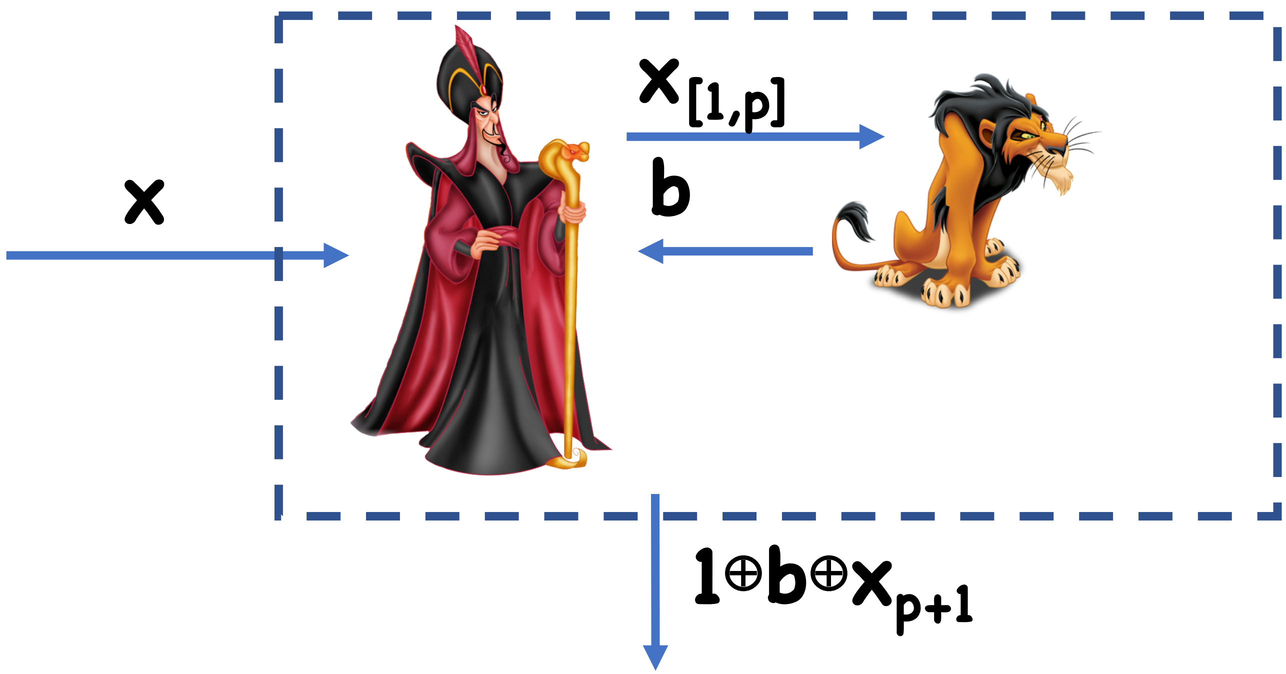
Assume towards contradiction  s.t.

$\Big|$ $\Pr[G(s)_{p+1} \leftarrow$  $(G(s)_{[1,p]})\ ] - ½$ $\Big|$ $> \varepsilon$

Proof

Pseudorandomness → Unpredictability

Construct 



Proof

Pseudorandomness → Unpredictability

Analysis:

- If $\mathbf{x}$ is random, $\Pr[1 \oplus b \oplus x_{p+1} = 1] = ½$
- If $\mathbf{x}$ is pseudorandom,

$$\Pr[1 \oplus b \oplus x_{p+1} = 1]$$

$= \Pr[G(s)_{p+1} \leftarrow$  $(G(s)_{[1,p]})\ ]$

$> (½ + \varepsilon)$ or $< (½ - \varepsilon)$

Proof

Unpredictability → Pseudorandomness

Assume towards contradiction  s.t.

$\Big|$ Pr[ $(G(s))=1 : s \leftarrow \{0,1\}^{\lambda}]$

$-$ Pr[ $(x)=1 : x \leftarrow \{0,1\}^{t}]\ \Big| \ > \varepsilon$

Proof

Unpredictability → Pseudorandomness

Hybrids:

H_i: $x_{[1,i]} \leftarrow G(s)$, $x_{[i+1,t]} \leftarrow \{0,1\}^{t-i}$

H_0: truly random **x**

H_t: pseudorandom **t**

Proof

Unpredictability → Pseudorandomness

Hybrids:

H_i: $x_{[1,i]} \leftarrow G(s)$, $x_{[i+1,t]} \leftarrow \{0,1\}^{t-i}$

| Pr[ $(x)=1 : x \leftarrow H_s$]

– Pr[ $(x)=1 : x \leftarrow H_0$] | $> \varepsilon$

Let q_i = Pr[ $(x)=1 : x \leftarrow H_i$]

Proof

Unpredictability → Pseudorandomness

Hybrids:

H_i: $x_{[1,i]} \leftarrow G(s)$, $x_{[i+1,t]} \leftarrow \{0,1\}^{t-i}$

$$| q_t - q_0 | > \varepsilon$$

Let $q_i = \Pr[$  $(x)=1 : x \leftarrow H_i]$

Proof

Unpredictability → Pseudorandomness

Hybrids:

H_i: $x_{[1,i]} \leftarrow G(s)$, $x_{[i+1,t]} \leftarrow \{0,1\}^{t-i}$

By triangle inequality, there must exist an **i** s.t.

$$| q_i - q_{i-1} | > \varepsilon/t$$

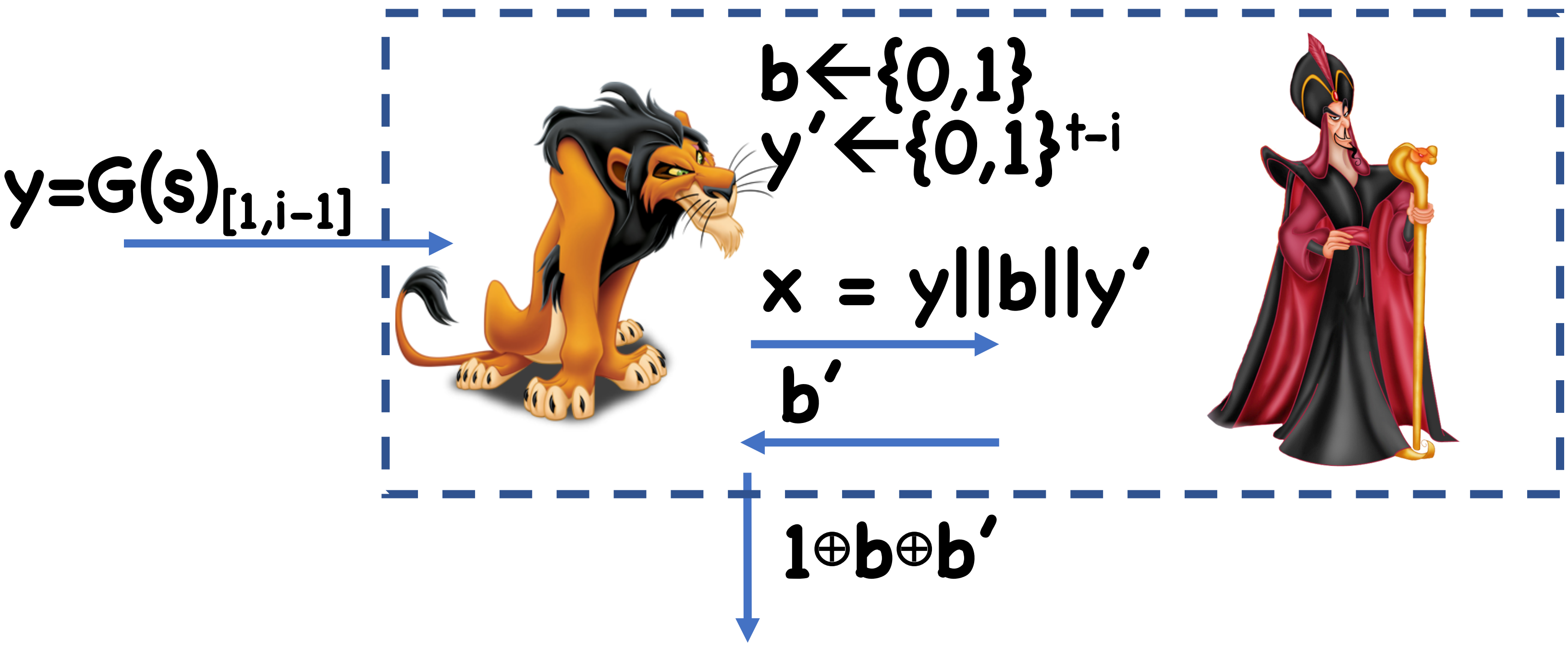
Can assume wlog that

$$q_i - q_{i-1} > \varepsilon/t$$

Proof

Unpredictability → Pseudorandomness

Construct 



Proof

Unpredictability → Pseudorandomness

Analysis:

- If $\mathbf{b} = \mathbf{G(s)}_i$, then  sees $\mathbf{H}_i$

  $\Rightarrow$  outputs **1** with probability $\mathbf{q}_i$

  $\Rightarrow$  outputs $\mathbf{b=G(s)}_i$ with probability $\mathbf{q}_i$

Proof

Unpredictability → Pseudorandomness

Analysis:

- If $\mathbf{b = 1 \oplus G(s)_i}$, then

  Define $\mathbf{q_i'}$ as $\mathbf{Pr[}$  outputs $\mathbf{1]}$

  $$\tfrac{1}{2}(q_i' + q_i) = q_{i-1} \Rightarrow q_i' = 2q_{i-1} - q_i$$

  $\Rightarrow$  outputs $\mathbf{G(s)_{[1,i]}}$ with probability

  $$1 - q_i' = 1 + q_i - 2q_{i-1}$$

Proof

Unpredictability → Pseudorandomness

Analysis:

- Pr[ outputs $G(s)_i$]

$$= \frac{1}{2}(q_i) + \frac{1}{2}(1 + q_i - 2q_{i-1})$$

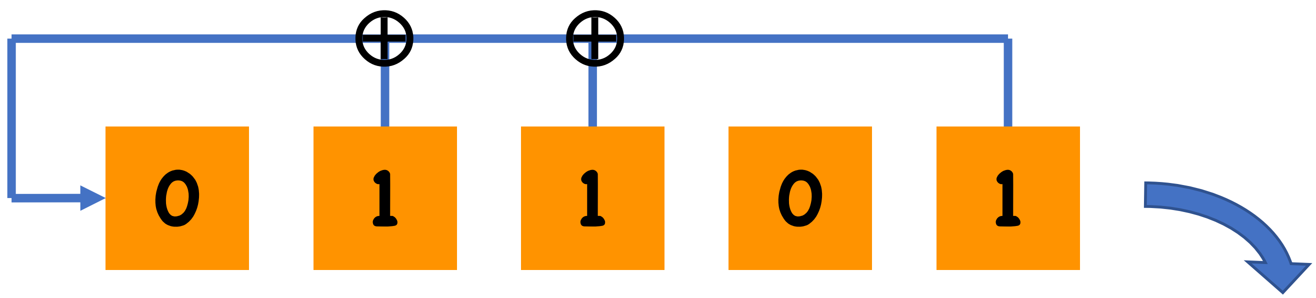
$$= \frac{1}{2} + q_i - q_{i-1}$$

$$> \frac{1}{2} + \varepsilon/t$$

Linearity

Linearity

LFSR's are linear:



$$\text{state}' = \begin{pmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix} \bullet \text{state}$$

$$\text{output} = (0\ 0\ 0\ 0\ 1) \bullet \text{state}$$

Linearity

LFSR's are linear:

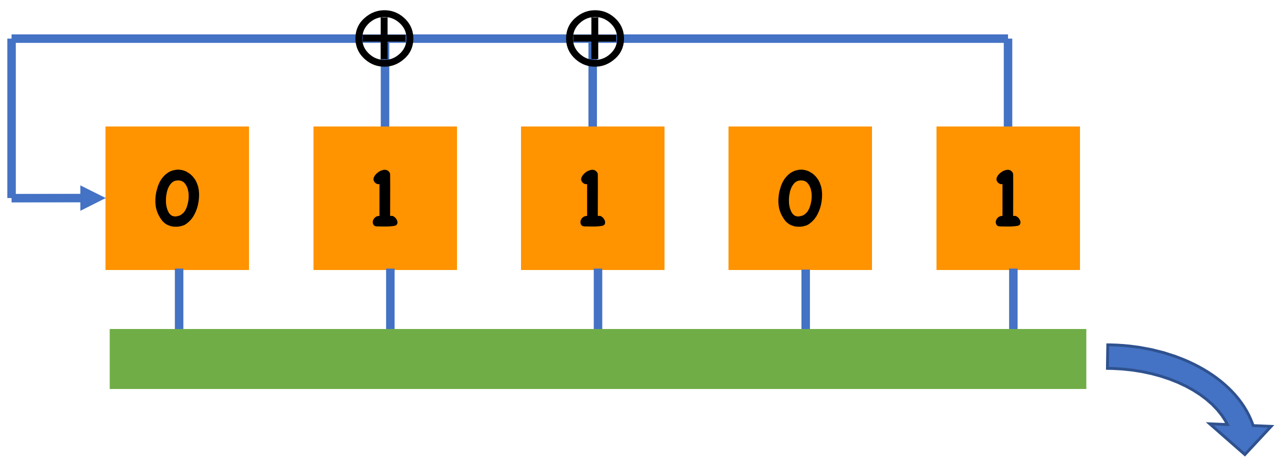
- Each output bit is a linear function of the initial state (that is, $G(s) = A \bullet s \pmod{2}$)

Any linear G <u>cannot</u> be a PRG

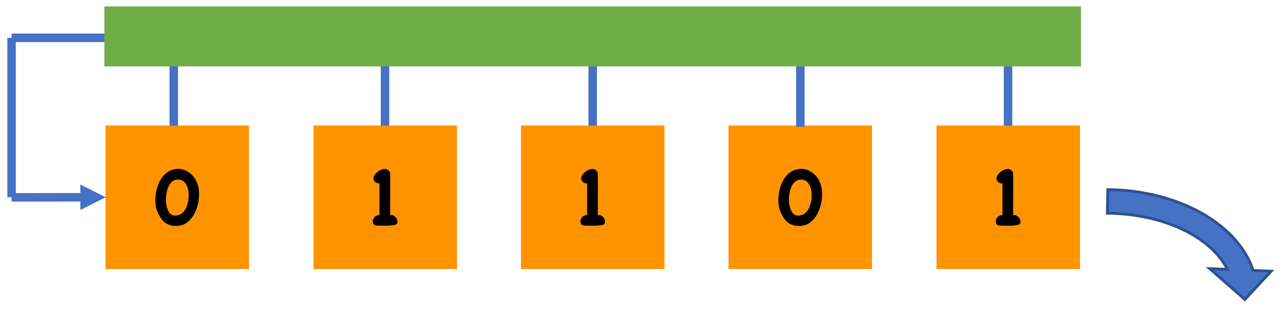
- Can check if x is in column-span of A using linear algebra

Introducing Non-linearity

Non-linearity in the output:



Non-linear feedback:



LFSR period

Period = number of bits before state repeats

After one period, output sequence repeats

Therefore, should have extremely long period

- Ideally almost 2^λ
- Possible to design LFSR's with period $2^\lambda-1$

Hardware vs Software

PRGs based on LFSR’s are very fast in hardware

Unfortunately, not easily amenable to software

RC4

Fast software based PRG

Resisted attack for several years

No longer considered secure, but still widely used

RC4

State = permutation on **[256]** plus two integers

- Permutation stored as **256**-byte array **S**

Init(16-byte **k)**:

- For **i=0,…,255**
  S[i] = i
- **j = 0**
- **For i=0,…,255**
  j = j + S[i] + k[i mod 16] (mod 256)
  Swap **S[i]** and **S[j]**
- Output **(S,0,0)**

RC4

GetBits(S,i,j):

- **i++ (mod 256)**
- **j+= S[i] (mod 256)**
- Swap **S[i]** and **S[j]**
- **t = S[i] + S[j] (mod 256)**
- Output **(S,i,j), S[t]**

New state — Next output byte

Insecurity of RC4

Second byte of output is slightly biased towards 0

- **Pr[**second byte **= 0^8] ≈ 2/256**
- Should be **1/256**

Means RC4 is not secure according to our definition

-  outputs **1** iff second byte is equal to **0^8**
- Advantage: **≈ 1/256**

Not a serious attack in practice, but demonstrates some structural weakness

Insecurity of RC4

Possible to extend attack to actually recover the input **k** in some use cases

- The seed is set to **(IV, k)** for some initial value **IV**
- Encrypt messages as **RC4(IV,k)**$\oplus$**m**
- Also give **IV** to attacker
- Cannot show security assuming RC4 is a PRG

Can be used to completely break WEP encryption standard

Extending the Stretch of a PRG

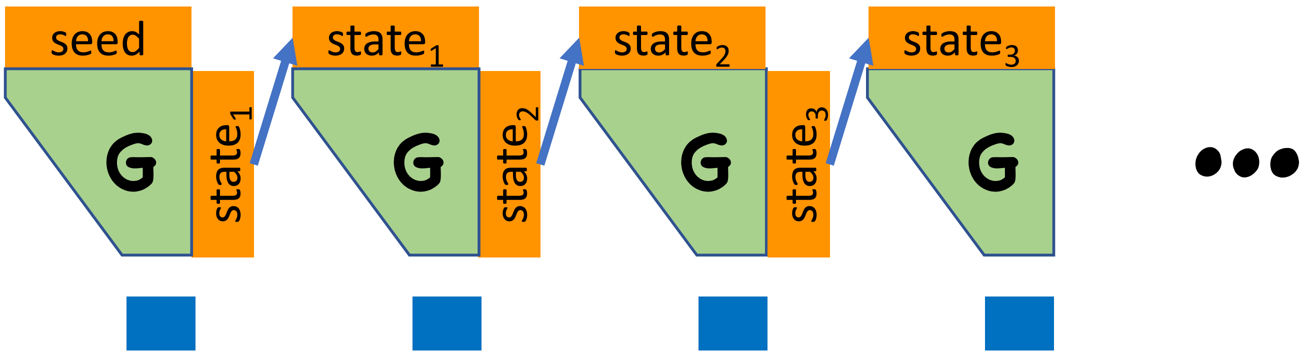
Suppose you have a fixed-stretch PRG **G**

- Better yet, a PRG that expands by a single bit

$$\mathbf{G}: \{0,1\}^{\lambda} \rightarrow \{0,1\}^{\lambda+1}$$

Construct a PRG **G'** of arbitrary output length

Extending the Stretch of a PRG

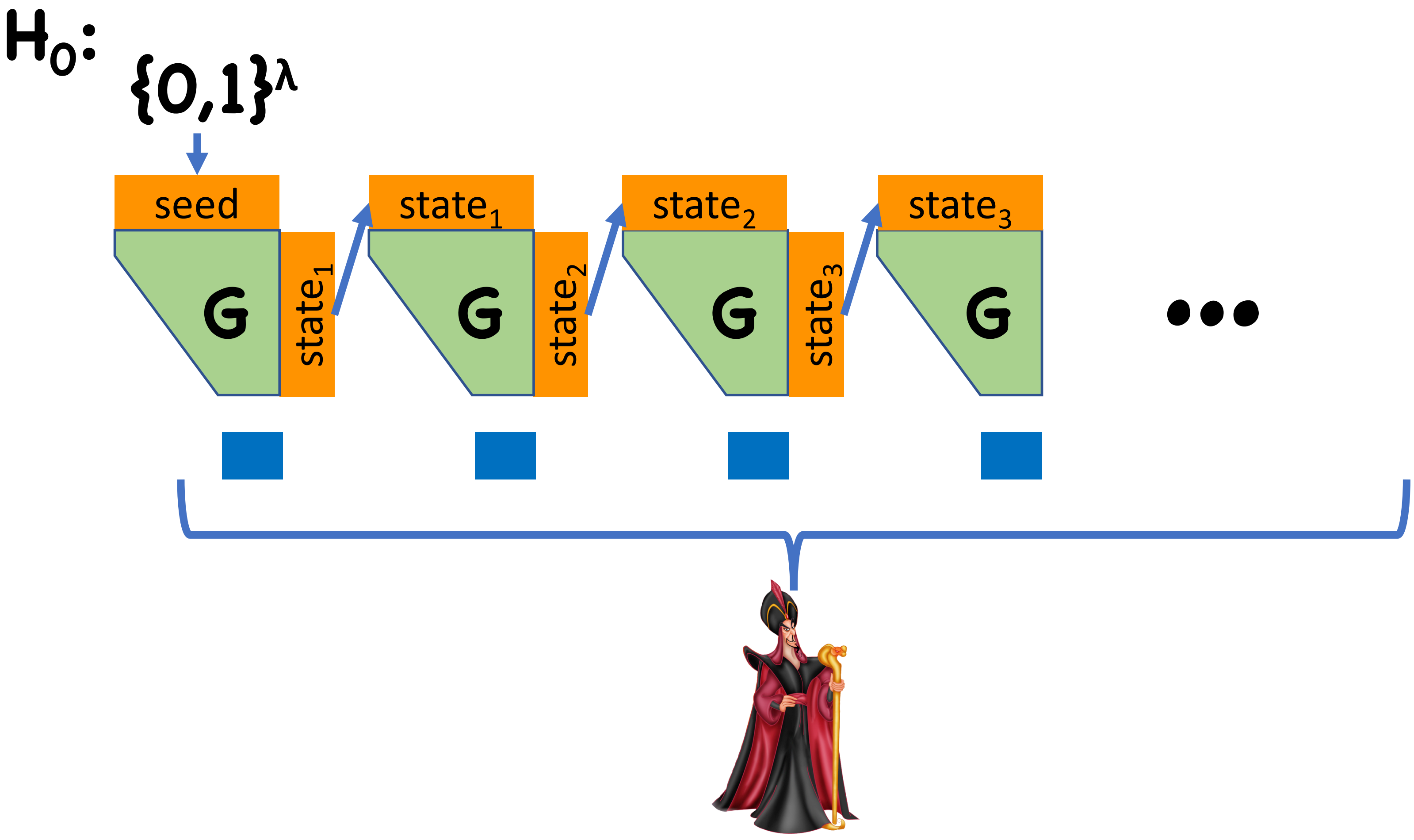


Security Proof

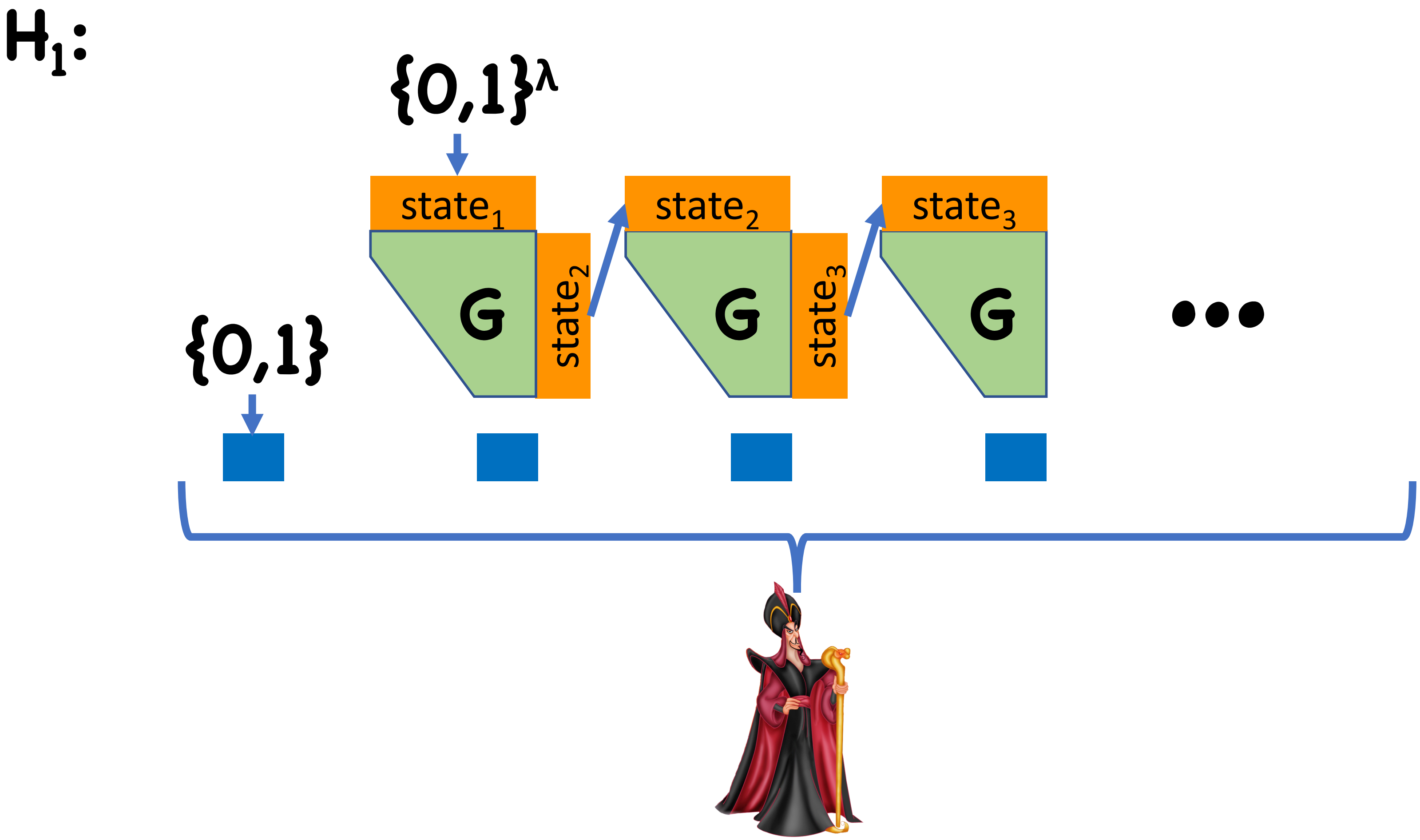
Assume towards contradiction  …

Define hybrids…

Security Proof

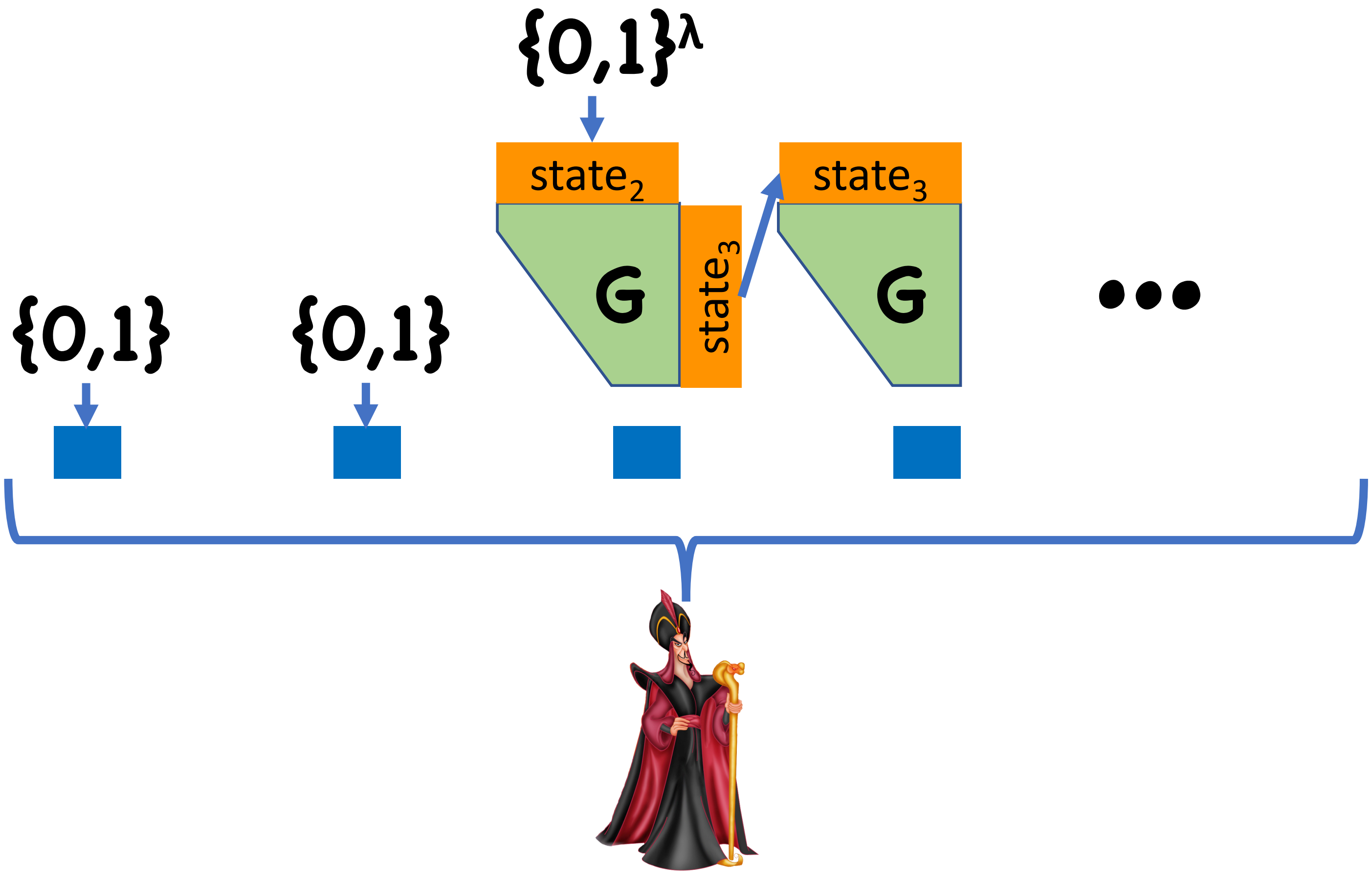


Security Proof



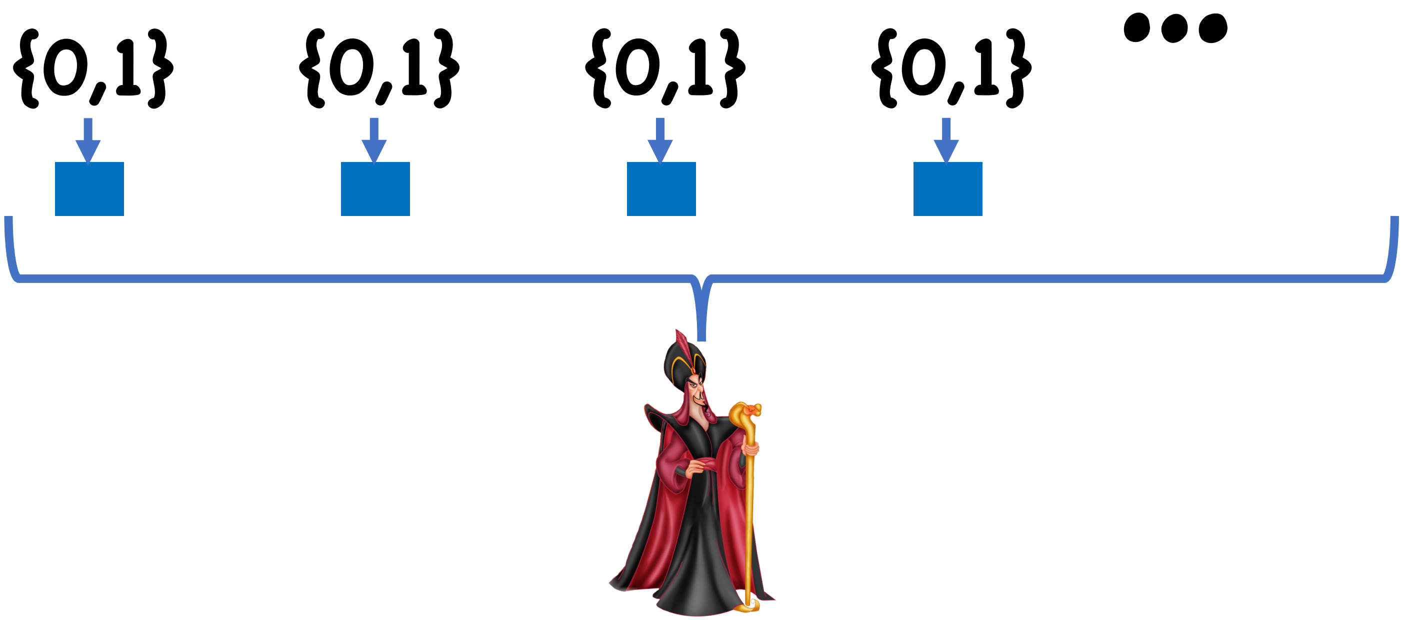
Security Proof

H_2:



Security Proof

H_t:



Security Proof

$\mathbf{H_0}$ corresponds to pseudorandom $\mathbf{x}$

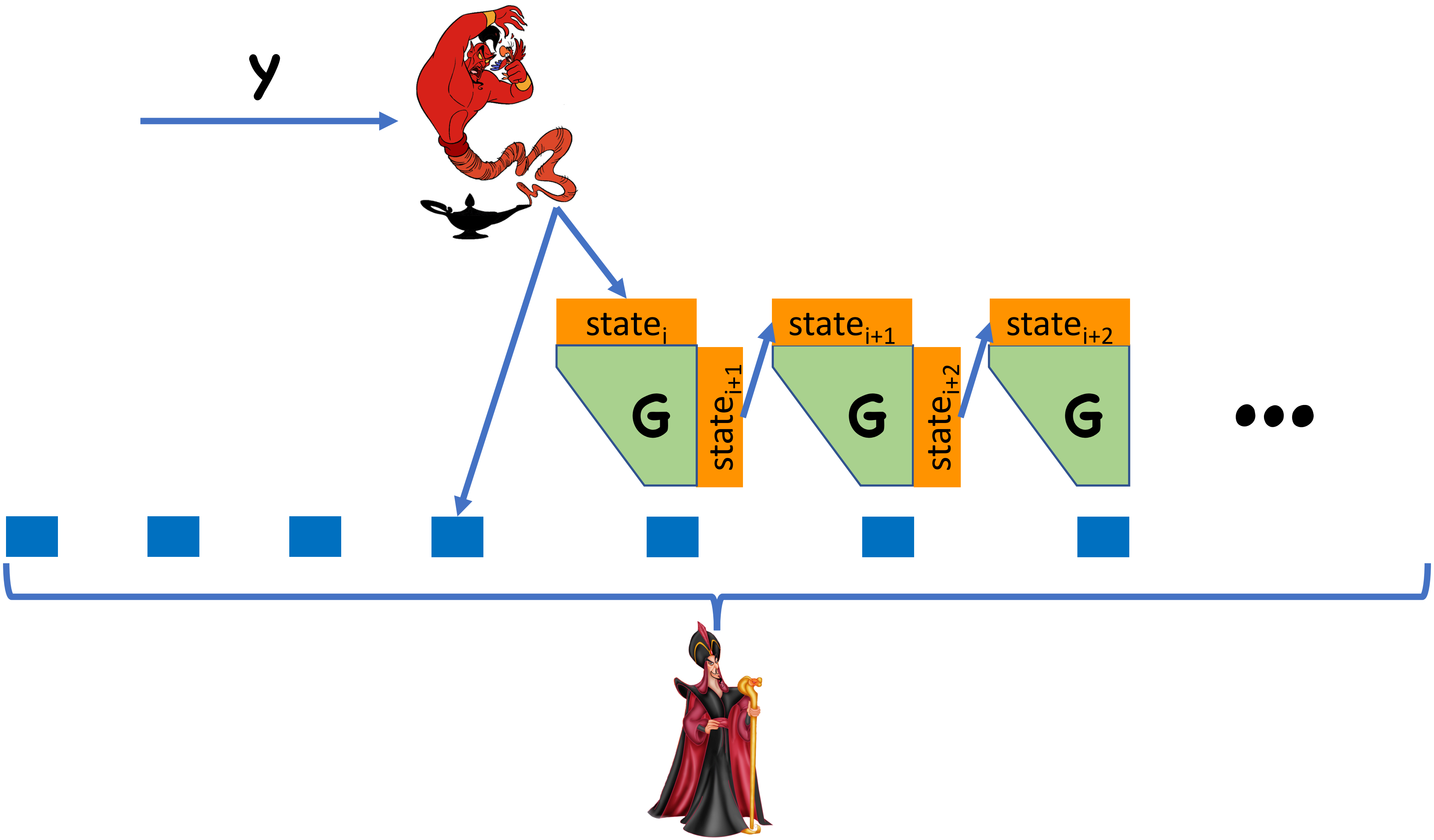
$\mathbf{H_t}$ corresponds to truly random $\mathbf{x}$

Let $\mathbf{q_i} = \mathbf{Pr[}$  $\mathbf{(x)=1 : x \leftarrow H_i\ ]}$

By assumption, $\mathbf{|q_t - q_0| > \varepsilon}$

$\Rightarrow\ \exists \mathbf{i}$ s.t. $\mathbf{|q_i - q_{i-1}| > \varepsilon/t}$

Security Proof



Security Proof

Analysis

- If $\mathbf{y} = \mathbf{G(s)}$, then  sees $\mathbf{H}_{i-1}$

  $\Rightarrow$ **Pr[**  **outputs 1] =** q_{i-1}

  $\Rightarrow$ **Pr[**  **outputs 1] =** q_{i-1}

- If $\mathbf{y}$ is random, then  sees $\mathbf{H}_i$

  $\Rightarrow$ **Pr[**  **outputs 1] =** q_i

  $\Rightarrow$ **Pr[**  **outputs 1] =** q_i

Summary

Stream ciphers = secure encryption for arbitrary length, number of messages
(though we did not completely prove it)

However, implementation difficulties due to having to maintaining state

Reminders

Project 1 part 1 Due Tomorrow

HW2 will be released tonight