COMS BC3262: Introduction to Cryptography

Lecture 6: Modes of Operation and OWFs

February 9, 2026

Logistics

Office hours:

- **Eysa**: Wed 3-5 this week (normally Mondays 3-5), Milstein 512
- **Mark**: Tuesdays 6:30-8:30, Milstein 503

If you can’t make either of these times, you can contact about additional office hours

PS1 was due last week, but late deadline has been extended to Thursday

Each late day is 10% off, but we’ll cap the late deduction at 30%

PS2 is due next week Thursday

Today’s Lecture

- Review
    - HW and definitions we’ve seen
- Block ciphers
    - PRP review
    - Modes of Operation (continued)
- Back to theory land
    - One-way functions

Tips for Proving Security (or Insecurity)

1. Write down your definition of security
    - For example, if something asks to show something is or is not EAV-secure, remind yourself what EAV-security looks like
2. Take note of the “for all” and “there exists”
    - To show something does **not** hold a “for all”, you need to show the **existence** of a **counterexample** for which the statement does not hold
        - For example, to show something is *not* CPA-secure, what messages could an adversary send to be able to distinguish the challenge messages? Analyze the success probability of the adversary winning

Tips for Proving Security (or Insecurity)

3. When working with computational assumptions, we will typically prove security via a **reduction**

   - We're essentially doing a proof by contradiction:

     "Suppose for contradiction X is not secure. If that were the case, then there would exist some adversary A that could break the security of X. If such an adversary A existed, we could use A to construct an adversary B to break the security of some cryptographic primitive Y. However, if we assume Y is secure, then such a B cannot exist. Therefore, A cannot exist, and X is secure."

4. If you're proving something holds "for all", it is *not* sufficient to show that the property holds for one specific example

   - For example, if we're making a statement about n-bit messages, it's not sufficient to just show something works for 0^n and then be done. You need to show it holds for *all* messages.

Recall: Perfect Secrecy

Definition: A symmetric-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is perfectly secret if for every distribution M over $\mathscr{M}$, for all $m \in \mathscr{M}$, and for all $c \in \mathscr{C}$ such that $\Pr[\mathsf{Enc}(k, m) = c] > 0$ it holds that

$$\Pr_{M \leftarrow \mathscr{M}, k \leftarrow \mathsf{Gen}}[M = m \,|\, \mathsf{Enc}(k, m) = c] = \Pr_{M \leftarrow \mathscr{M}}[M = m]$$

Recall: Another Definition of Perfect Secrecy

Alternative Definition: A symmetric-key encryption scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is perfectly secret if for every $m_0, m_1 \in \mathscr{M}$ and for every $c \in \mathscr{C}$ it holds that

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

Theorem: These two definitions of perfect secrecy are equivalent

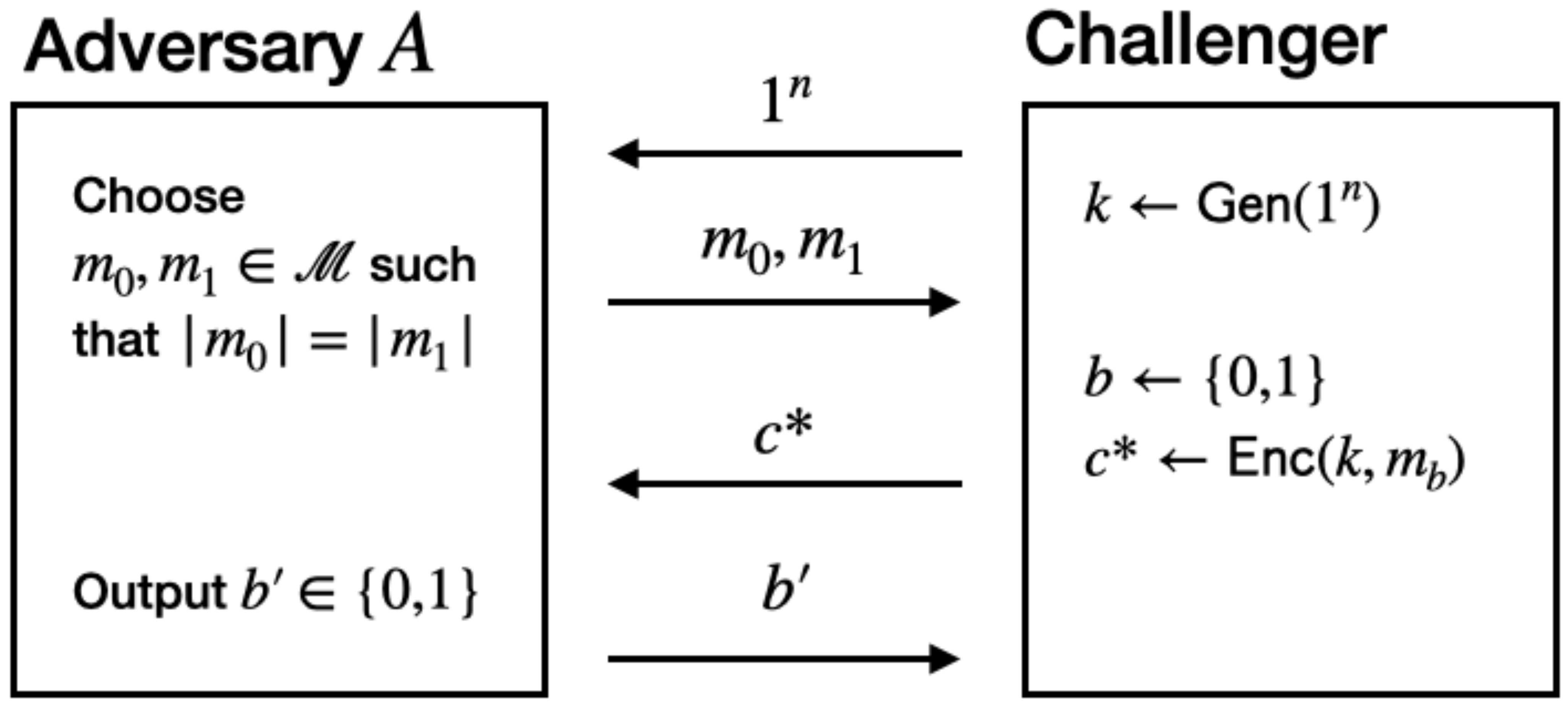
Recall: Indistinguishable Encryptions
(aka Semantic Security)

Given $\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ and an adversary A, consider the experiment $\text{PrivK}^{\text{eav}}_{\Pi,A}(n)$:

Definition:

Π has indistinguishable encryptions in the presence of an eavesdropper (EAV-security) if for every PPT adversary A there exists a negligible function $\epsilon(\cdot)$ such that

$$\Pr[\text{PrivK}^{\text{eav}}_{\Pi,A}(n) = 1] \leq \frac{1}{2} + \epsilon(n)$$



Adversary A

Choose $m_0, m_1 \in \mathcal{M}$ such that $|m_0| = |m_1|$

Output $b' \in \{0,1\}$

1^n

m_0, m_1

c^*

b'

Challenger

$k \leftarrow \text{Gen}(1^n)$

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

$c^* \leftarrow \text{Enc}(k, m_b)$

$\text{PrivK}^{\text{eav}}_{\Pi,A}(n) = 1$ if $b' = b$.

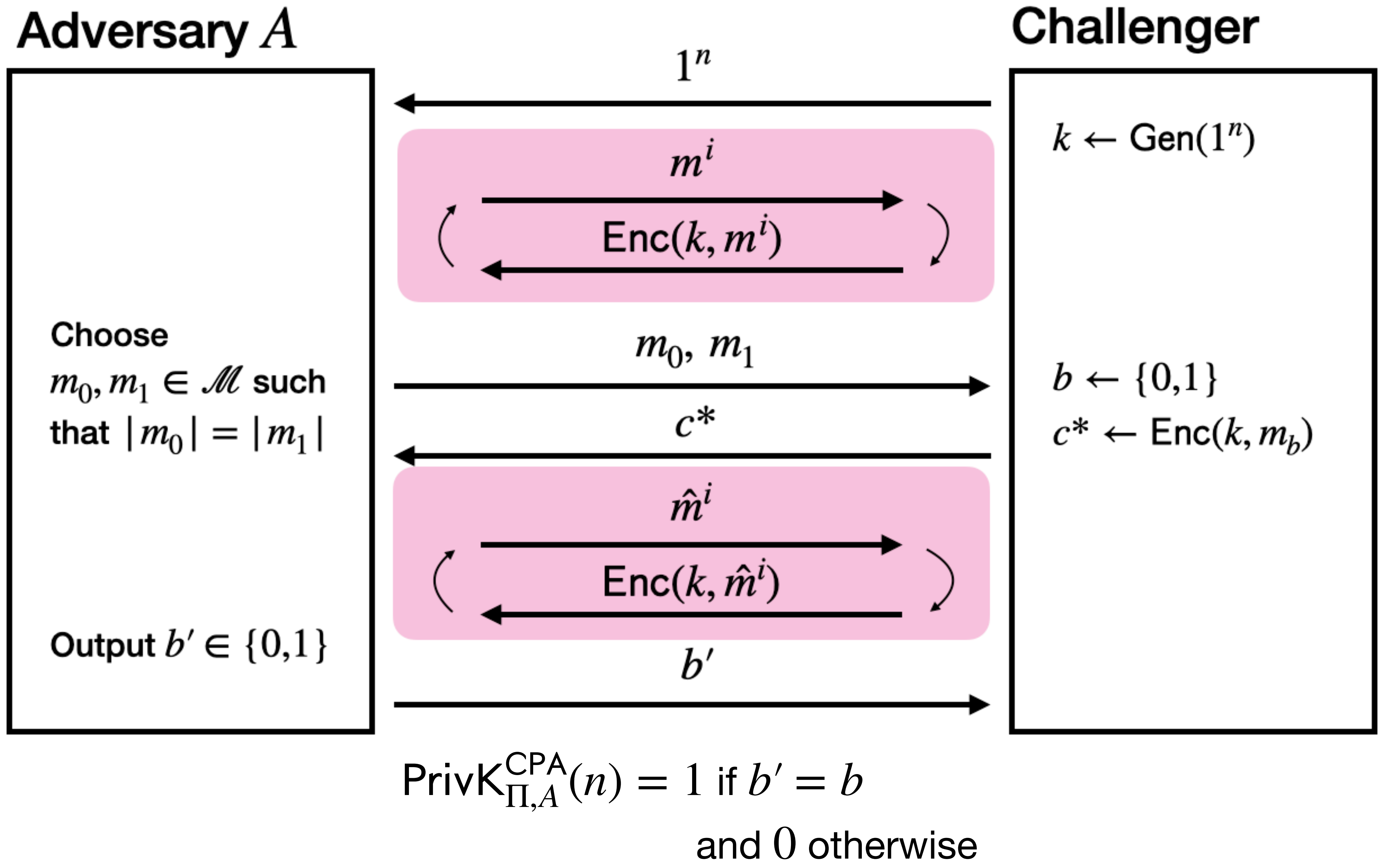
$\text{PrivK}^{\text{eav}}_{\Pi,A}(n) = 0$ otherwise

Recall: Chosen-Plaintext Attack (CPA)

Definition:

Π has indistinguishable encryptions under chosen-plaintext attack (or CPA-security) if for every PPT adversary A there exists a negligible function $\epsilon(\cdot)$ such that

$$\Pr[\mathsf{PrivK}^{\mathsf{CPA}}_{\Pi,A}(n) = 1] \leq \frac{1}{2} + \epsilon(n)$$



Adversary A
Challenger
1^n
$k \leftarrow \mathsf{Gen}(1^n)$
m^i
$\mathsf{Enc}(k, m^i)$
Choose $m_0, m_1 \in \mathcal{M}$ such that $|m_0| = |m_1|$
$m_0,\ m_1$
$b \leftarrow \{0,1\}$
$c^* \leftarrow \mathsf{Enc}(k, m_b)$
c^*
$\hat{m}^i$
$\mathsf{Enc}(k, \hat{m}^i)$
Output $b' \in \{0,1\}$
b'

$\mathsf{PrivK}^{\mathsf{CPA}}_{\Pi,A}(n) = 1$ if $b' = b$
and 0 otherwise

Notes:

- CPA Security implies multiple message security
- Any CPA secure private key encryption scheme must have a **randomized** encryption algorithm

Recall: Pseudorandom Generators (PRGs)

Definition: Let G be a deterministic polynomial-time algorithm and $\ell(\cdot)$ be a polynomial s.t. for any input $s \in \{0,1\}^n$ we have $G(s) \in \{0,1\}^{\ell(n)}$. Then G is a **pseudorandom generator** if the following two conditions hold:

- **Expansion**: $\ell(n) > n$
- **Pseudorandomness**: For every PPT "distinguisher" D there exists a negligible function negl$(\cdot)$ s.t.

$$\left| \Pr_{s \leftarrow \{0,1\}^n}\left[D\left(G(s)\right) = 1\right] - \Pr_{r \leftarrow \{0,1\}^{\ell(n)}}\left[D(r) = 1\right] \right| \leq \text{negl}(n)$$

PRG or not?

Let G_1 and G_2 be length-doubling PRGs.

Are the following candidate constructions for G necessarily PRGs?

1. $G(s_1 \ldots s_n) = G_1(s_1 \ldots s_n) \,||\, G_2(s_1 \ldots s_n)$

2. $G(s_1 \ldots s_n) = G_1(s_1 \ldots s_m) \,||\, G_2(s_{m+1} \ldots s_n)$ where $m = \lfloor n/2 \rfloor$

3. $G'(s) = \begin{cases} G(s) & s \neq 0 \\ 0 \ldots 0 & s = 0 \end{cases}$

Pseudorandom Functions (PRFs)

Definition:

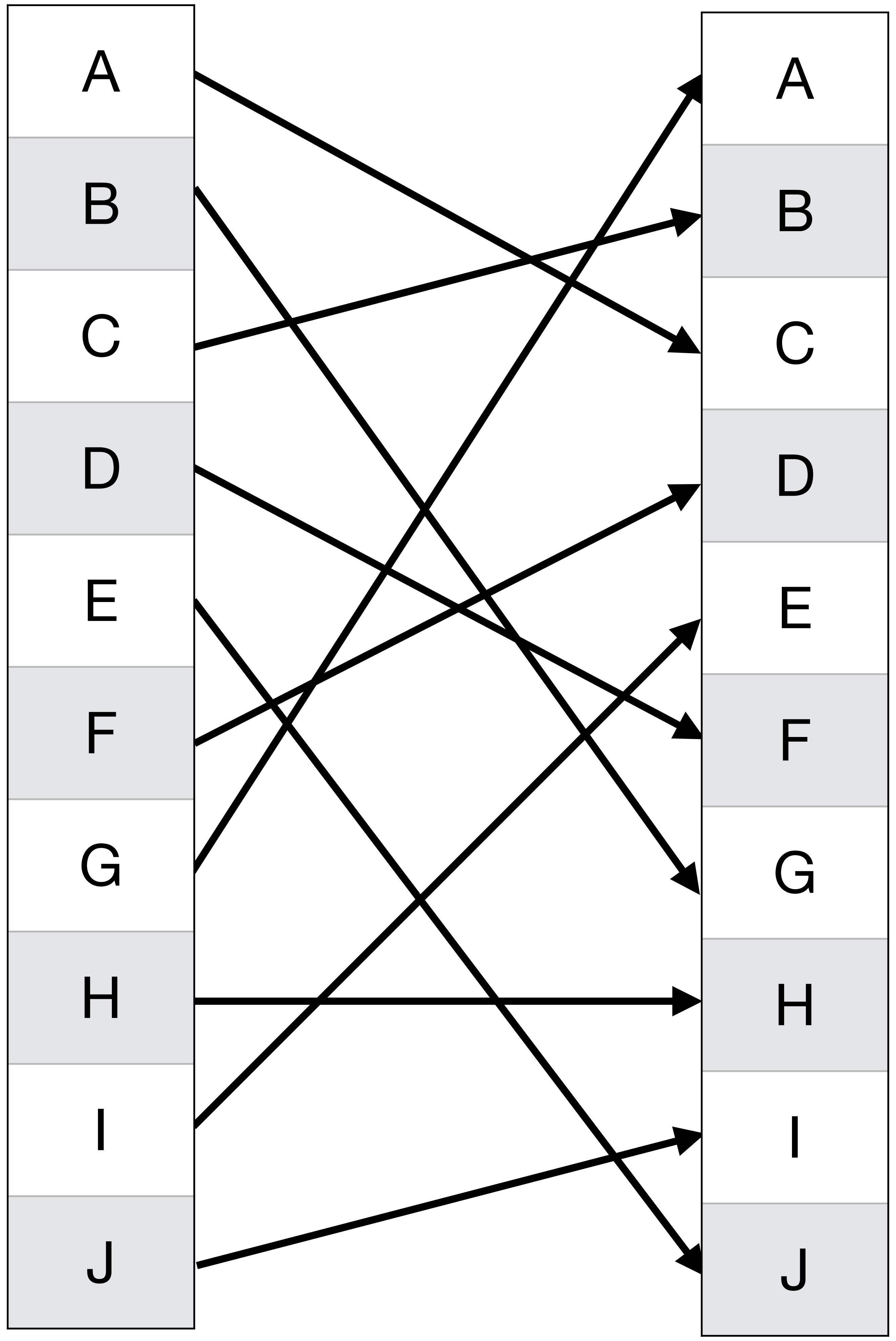
Let $F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ be an efficient, length-preserving, keyed function. We say that F is a pseudorandom function (PRF) if for all PPT D there exists a negligible function $\epsilon(\cdot)$ such that

$$\left| \Pr_{k \leftarrow \{0,1\}^n}\left[D^{F_k}\left(1^n\right) = 1\right] - \Pr_{f \leftarrow \mathscr{F}_n}\left[D^{f}\left(1^n\right) = 1\right] \right| \leq \epsilon(n)$$

Last time:
Pseudorandom Permutations (PRPs)

Permutations

- A permutation is a function f from a finite set to itself that is 1-1 and onto (bijection)
    - The inverse f^{-1} is well defined
- We will consider permutations over $\{0,1\}^{\ell}$
- Recall $\mathscr{F}_n$ is the set of all possible functions from $\{0,1\}^n \rightarrow \{0,1\}^n$
    - We will denote $\mathscr{P}_n$ as the set of all possible permutations from $\{0,1\}^n \rightarrow \{0,1\}^n$



Pseudorandom Permutations (PRPs)

Definition (PRP):

F is a pseudorandom permutation (PRP) if F is a PRF and

for all $k \in \{0,1\}^n$, $F_k : \{0,1\}^{\ell(n)} \to \{0,1\}^{\ell(n)}$ is a bijection (i.e., F_k^{-1} exists),

and F_k^{-1} is efficiently computable given k

(i.e., there is a polynomial-time algorithm that given k, y computes $F_k^{-1}(y)$)

Strong Pseudorandom Permutations (PRPs)

Definition (*Strong* PRP):

F is a *strong* pseudorandom permutation if F is a PRP and for all PPT D there exists a negligible function $\epsilon(\cdot)$ such that

$$\left| \Pr_{k\leftarrow\{0,1\}^n}\left[D^{F_k,F_k^{-1}}\left(1^n\right)=1\right] - \Pr_{f\leftarrow\mathscr{P}_n}\left[D^{f,f^{-1}}\left(1^n\right)=1\right]\right| \le \epsilon(n)$$

Pseudorandomness holds even if the distinguisher has access to F_k and F_k^{-1}

Modes of Operation

Practical Heuristics: Block Ciphers

- Block ciphers typically refers to practical constructions of strong PRPs
  - Described in terms of *concrete* values rather than asymptotics (i.e., input and output length are fixed rather than a function of the security parameter)
  - In practice, PRFs/PRPs in protocols are instantiated with block ciphers

Electronic CodeBook (ECB) Mode

- Naive mode of operation

- Just apply the block cipher to each block separately:

$$c = F_k(m_1), F_k(m_2), \ldots, F_k(m_\ell)$$

Electronic CodeBook (ECB) Mode

- Naive mode of operation

- Just apply the block cipher to each block separately:

$$c = F_k(m_1), F_k(m_2), \ldots, F_k(m_\ell)$$

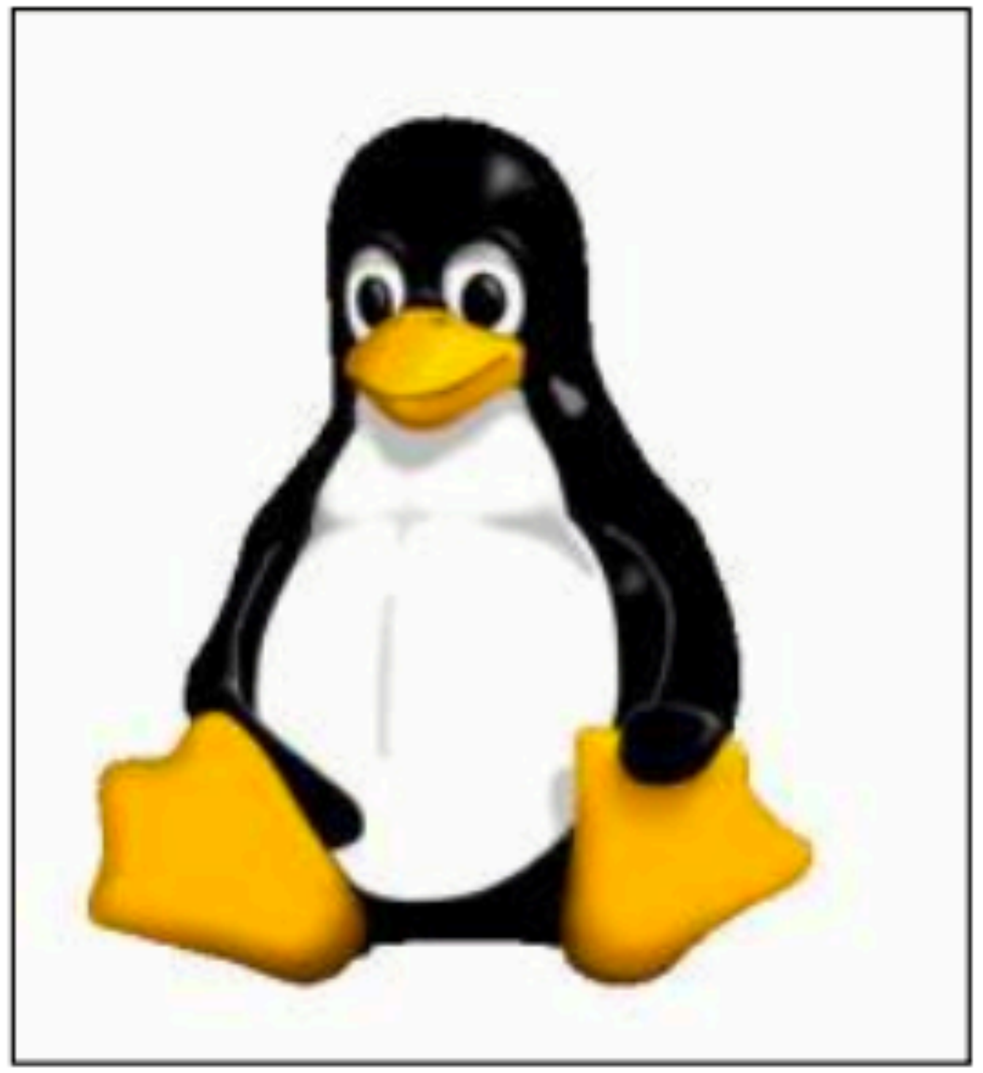
- … It should be very obvious that this is not good

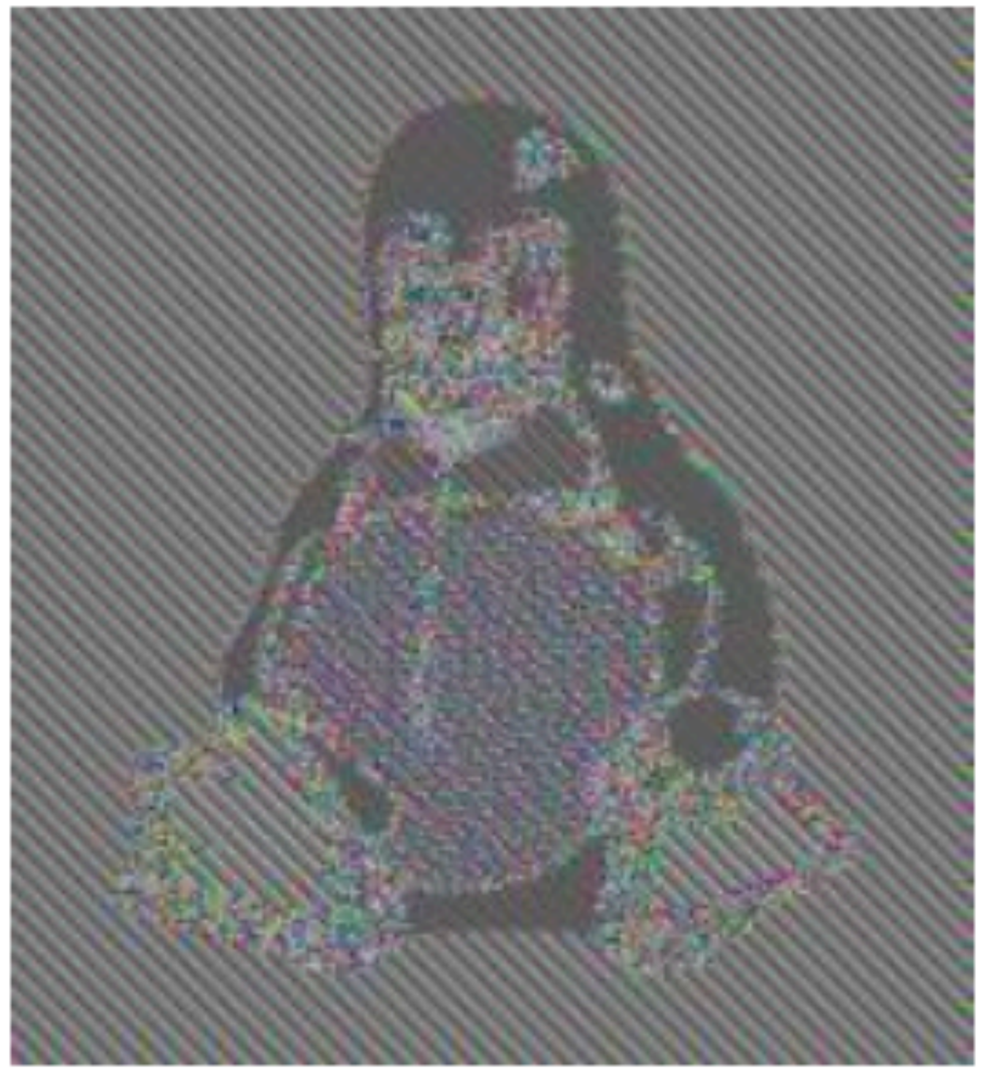
Electronic CodeBook (ECB) Mode

$$c = F_k(m_1), F_k(m_2), \ldots, F_k(m_\ell)$$

- Example by penguin picture:

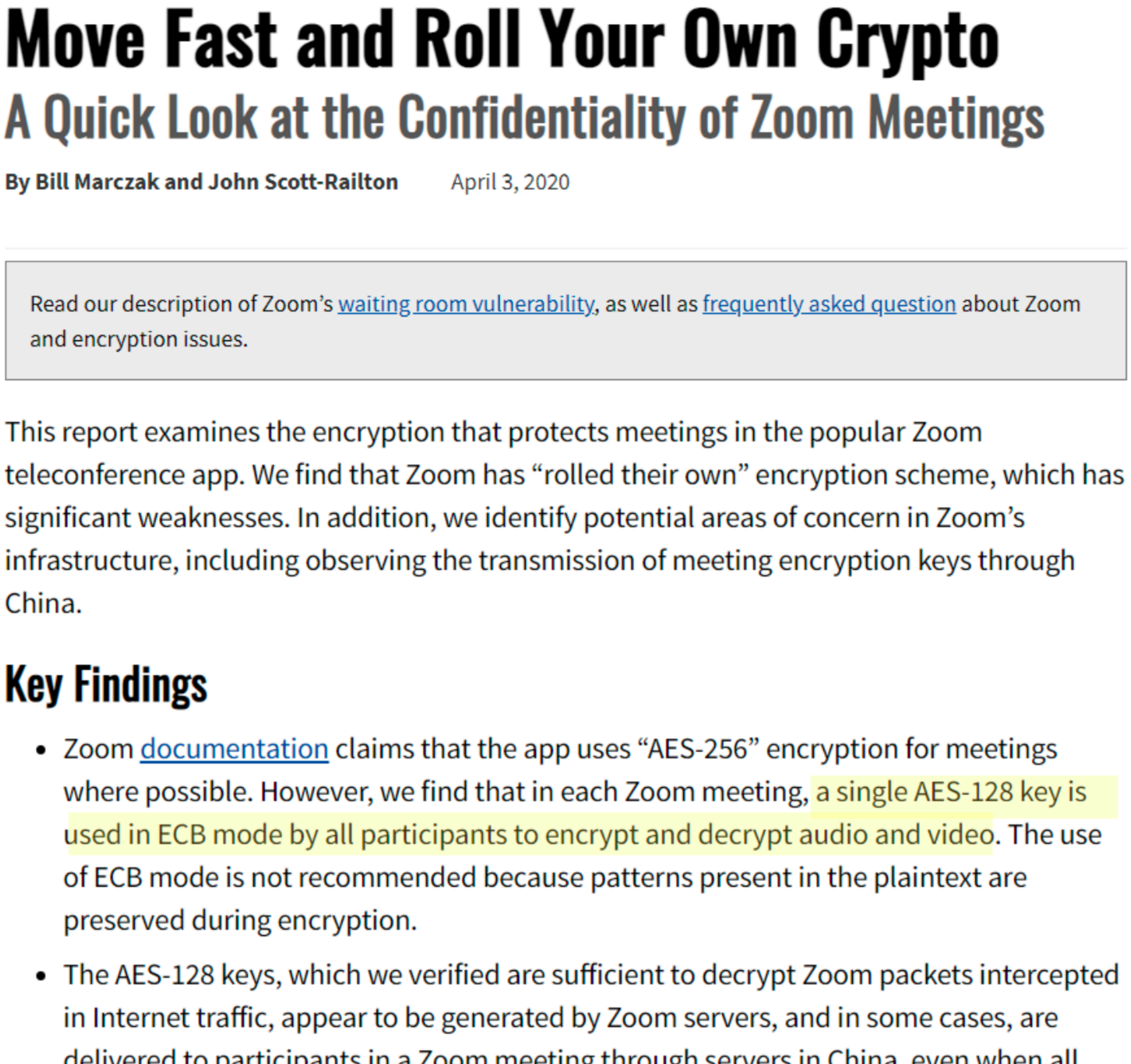
Original image





ECB mode encryption

https://en.wikipedia.org/wiki/Block_cipher_mode_of_operation



Move Fast and Roll Your Own Crypto

A Quick Look at the Confidentiality of Zoom Meetings

By Bill Marczak and John Scott-Railton April 3, 2020

Read our description of Zoom's waiting room vulnerability, as well as frequently asked question about Zoom and encryption issues.

This report examines the encryption that protects meetings in the popular Zoom teleconference app. We find that Zoom has "rolled their own" encryption scheme, which has significant weaknesses. In addition, we identify potential areas of concern in Zoom's infrastructure, including observing the transmission of meeting encryption keys through China.

Key Findings

- Zoom documentation claims that the app uses "AES-256" encryption for meetings where possible. However, we find that in each Zoom meeting, a single AES-128 key is used in ECB mode by all participants to encrypt and decrypt audio and video. The use of ECB mode is not recommended because patterns present in the plaintext are preserved during encryption.
- The AES-128 keys, which we verified are sufficient to decrypt Zoom packets intercepted in Internet traffic, appear to be generated by Zoom servers, and in some cases, are delivered to participants in a Zoom meeting through servers in China, even when all

Cipher Block Chaining (CBC) Mode

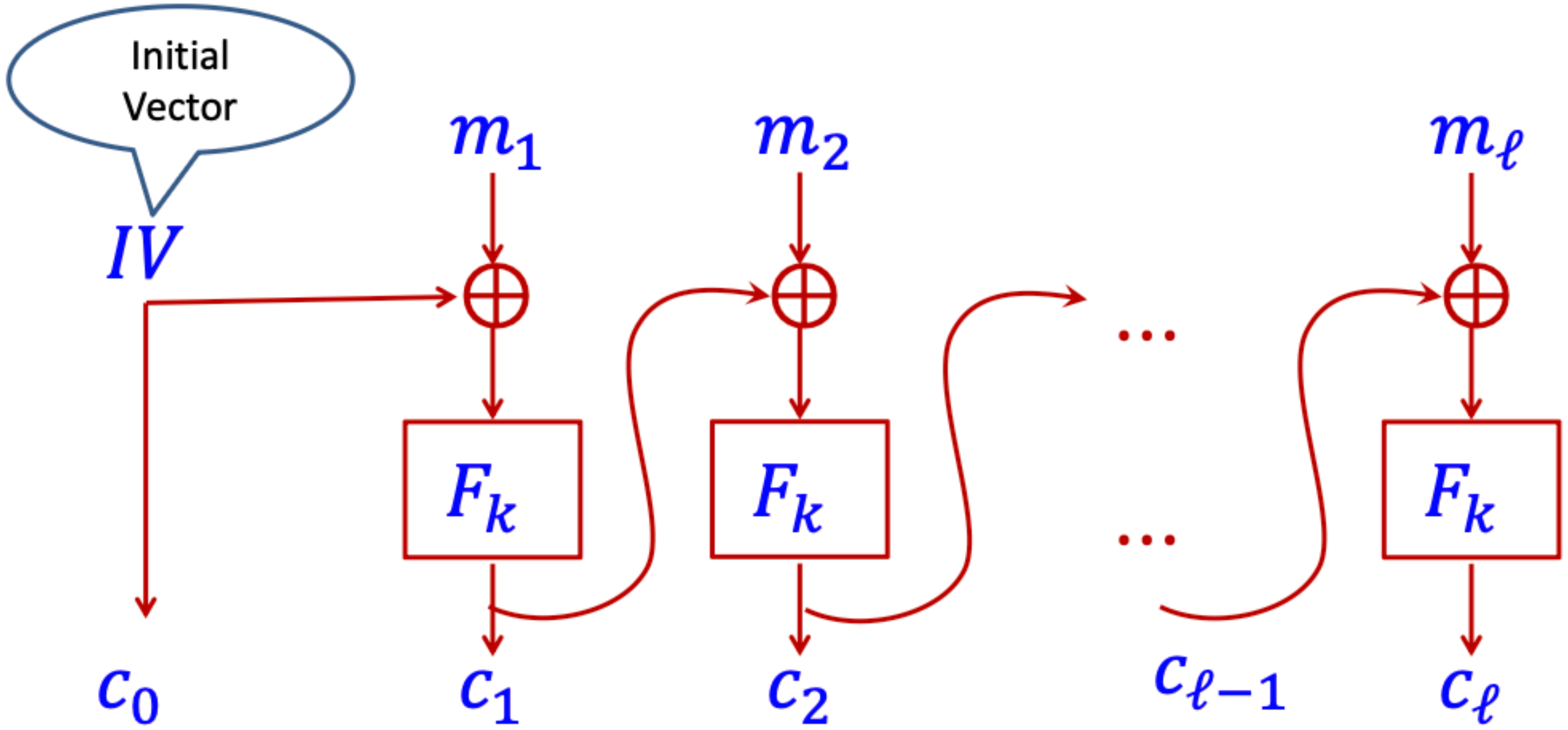
- This mode starts by sampling a random “initialization vector” (IV) and setting that as the first part of the cipher text.

- Each next part of the cipher text is obtained by applying the block cipher to the XOR of the previous cipher text and the message:

$$\begin{aligned} c &= (c_0, c_1, c_2, \ldots, c_\ell) \\ &= (IV, F_k(c_0 \oplus m_1), F_k(c_1 \oplus m_2), \ldots, F_k(c_{\ell-1} \oplus m_\ell)) \end{aligned}$$

Cipher Block Chaining (CBC) Mode

$$\begin{aligned} c &= (c_0, c_1, c_2, \ldots, c_\ell) \\ &= (IV, F_k(c_0 \oplus m_1), F_k(c_1 \oplus m_2), \ldots, F_k(c_{\ell-1} \oplus m_\ell)) \end{aligned}$$



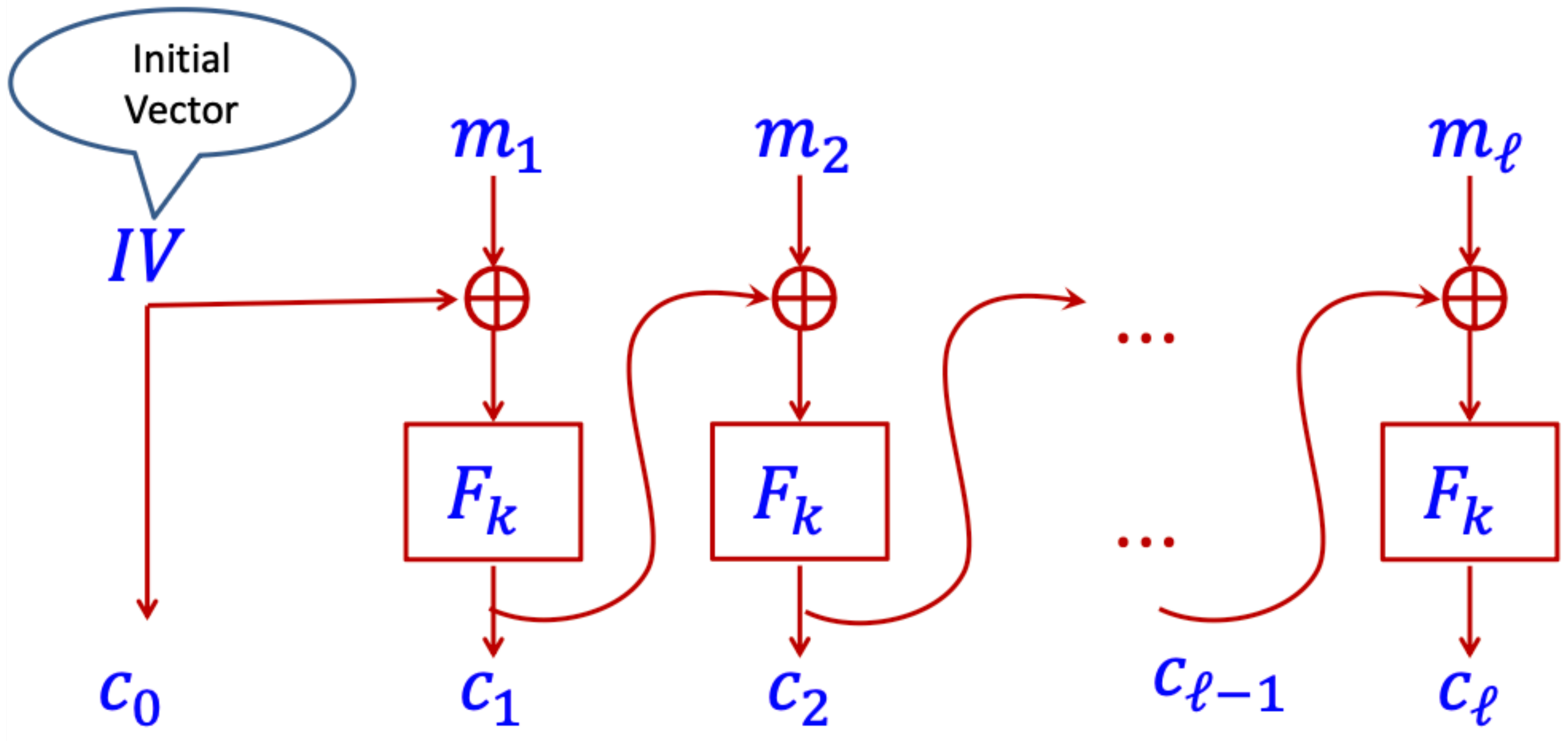
CBC Properties

- **Theorem**: If F is a strong PRP, then CBC is CPA-secure

- Expansion is only one block (compared to applying our OTP-inspired technique which doubled the size)

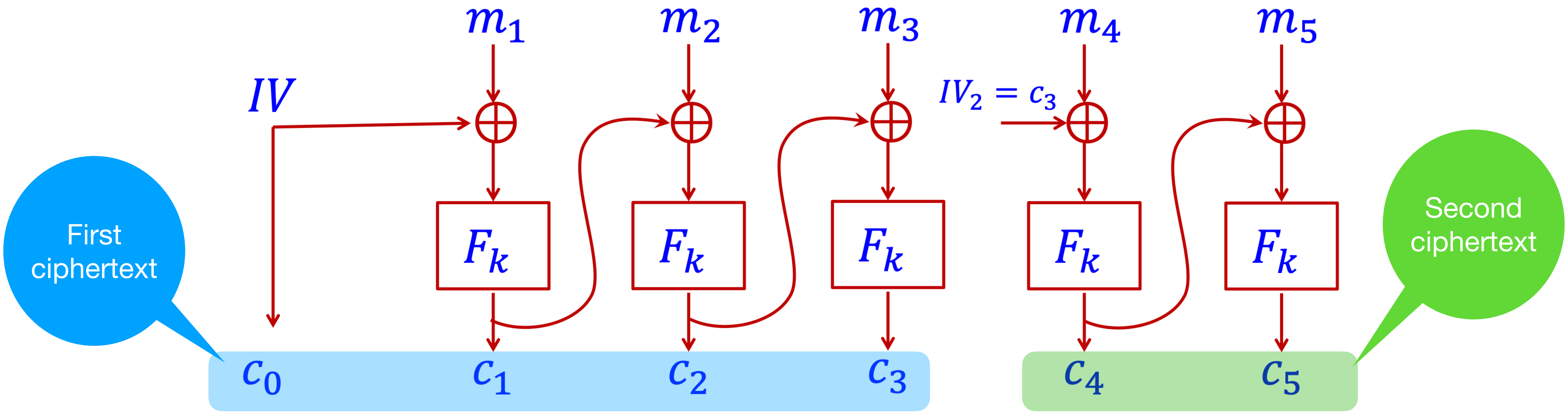
- Encryption is inherently **sequential** and cannot be done in parallel

- Message length is assume to be a multiple of block length, but padding can be added



Chained CBC

- Stateful variant of CBC that uses the last block of the previous ciphertext as the IV for the next encryption
    - Was used in SSL 3.0 and TLS 1.0
- Looks very similar to CBC… Is it also CPA-secure?



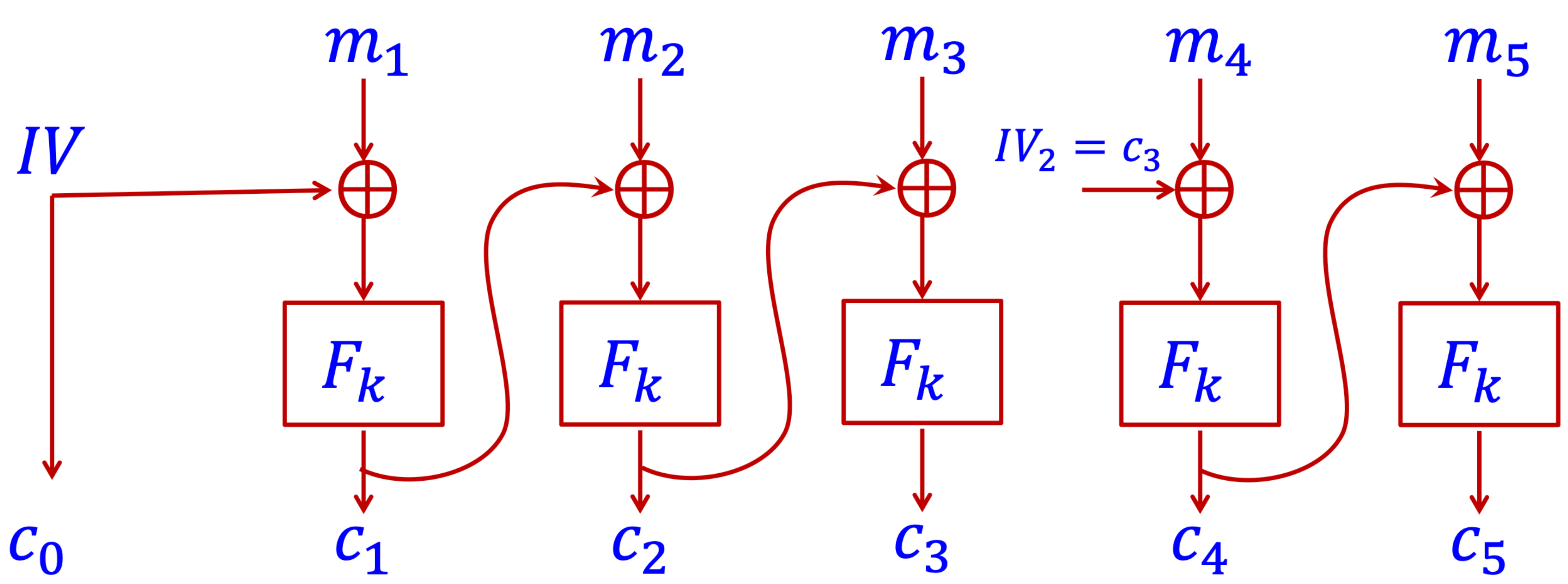
Chained CBC is not CPA-Secure!

IV_2 is known before (m_4, m_5) is chosen!

- An adversary who sees (c_0, c_1, c_2, c_3) may choose (m_4, m_5) based on this!

Attack based on this was discovered in 2002 but was considered theoretical

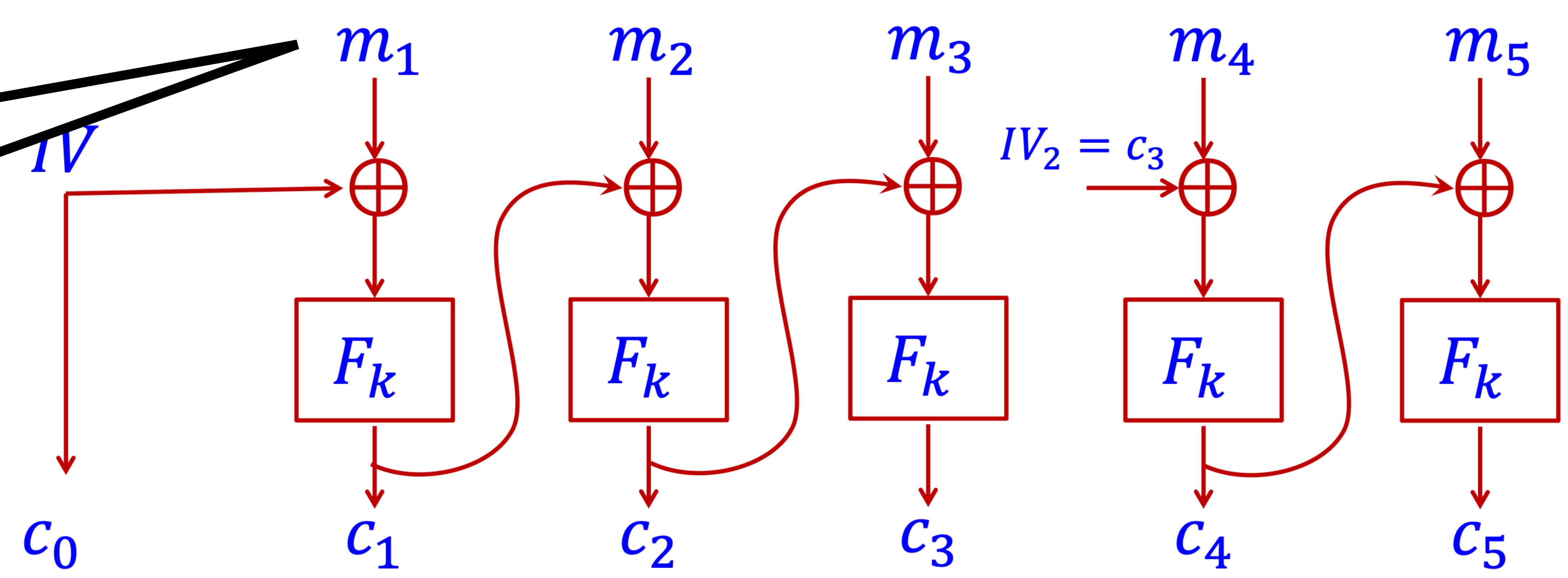
- 2011 it was demonstrated practically with a BEAST attack
  - BEAST attack did require some additional components like HTTP injections, but the main premise was with how TLS 1.0 and 1.1 did CBC



Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

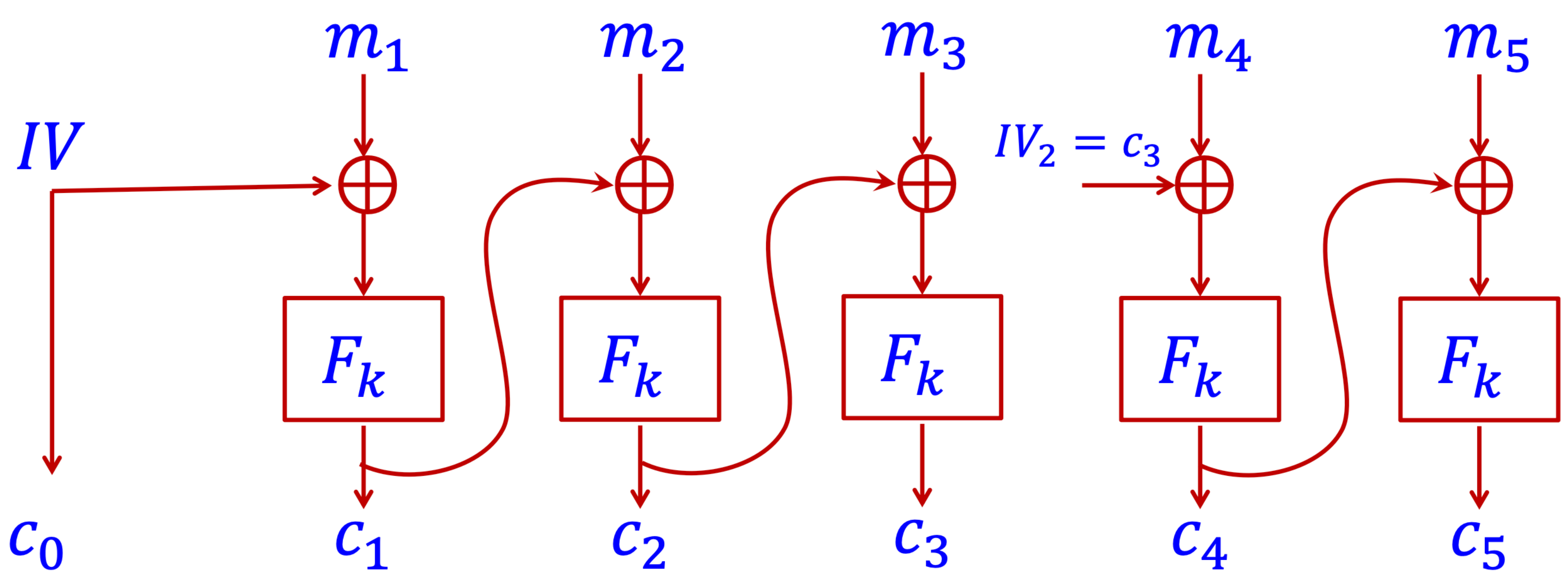
What if m_1= “POST /arbitrary prefix HTTP /1.1 password=X” and adversary was trying to guess what X is?



Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

Adversary sets $m_4 = c_3 \oplus IV \oplus m_1^a$



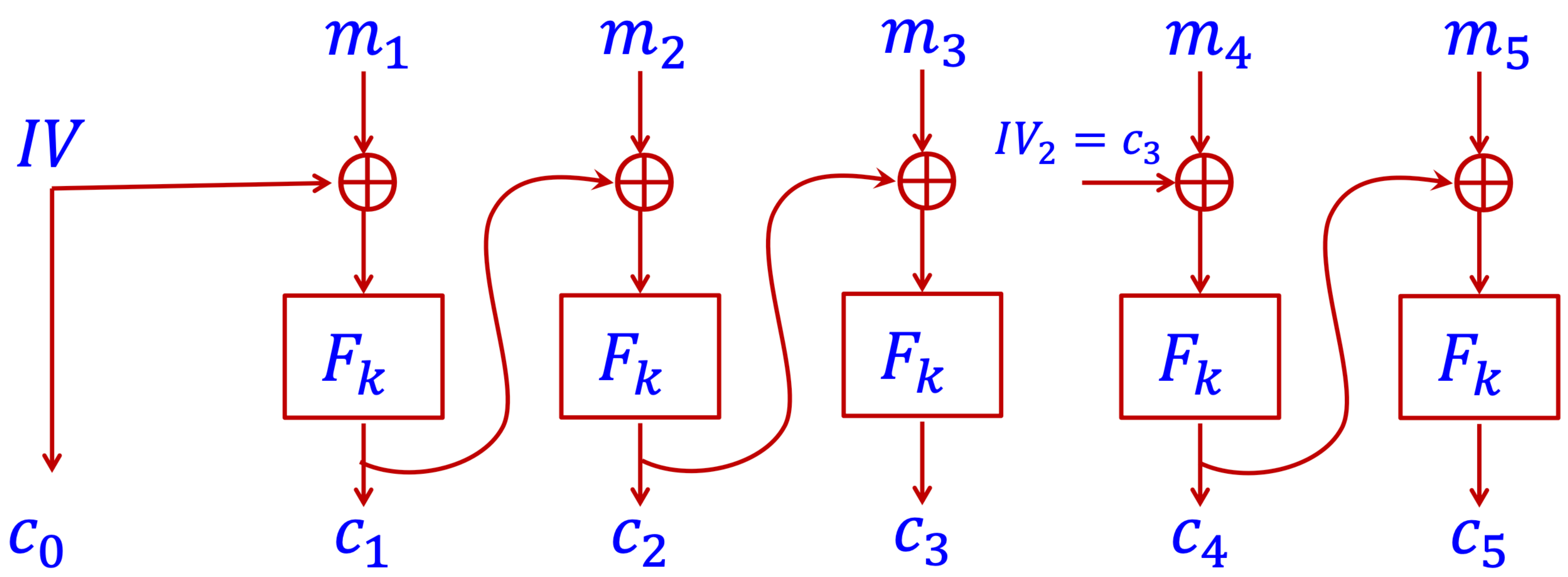
Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

Adversary sets $m_4 = c_3 \oplus IV \oplus m_1^a$

If m_1 was m_1^a :

$$c_4 = F_k(c_3 \oplus m_4)$$



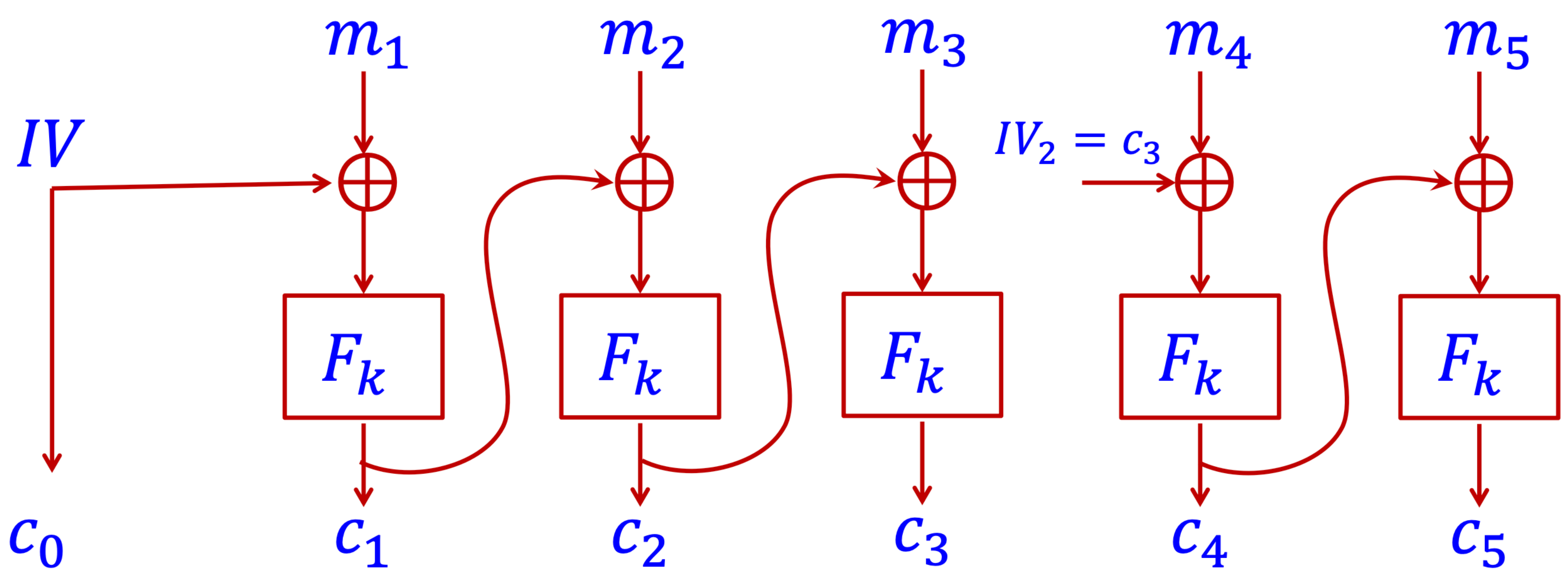
Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

Adversary sets $m_4 = c_3 \oplus IV \oplus m_1^a$

If m_1 was m_1^a :

$$c_4 = F_k(c_3 \oplus m_4)$$
$$= F_k(c_3 \oplus c_3 \oplus IV \oplus m_1)$$



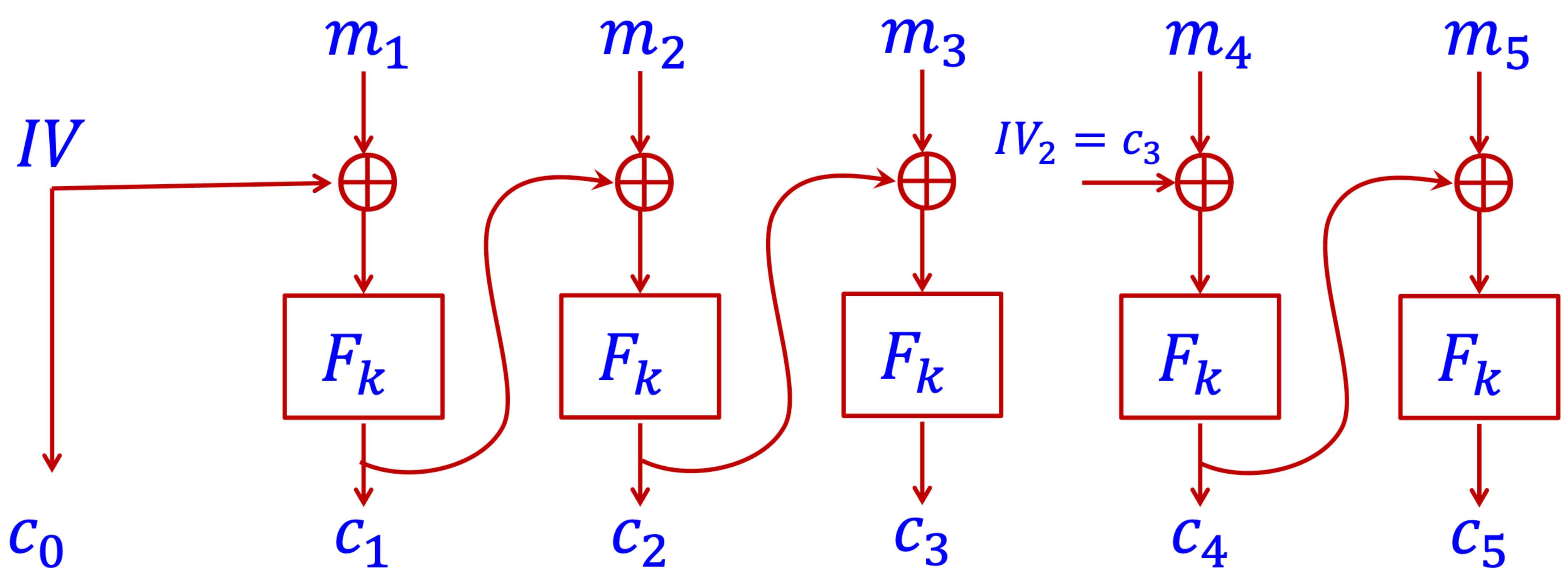
Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

Adversary sets $m_4 = c_3 \oplus IV \oplus m_1^a$

If m_1 was m_1^a :

$$\begin{aligned} c_4 &= F_k(c_3 \oplus m_4) \\ &= F_k(c_3 \oplus c_3 \oplus IV \oplus m_1) \\ &= F_k(IV \oplus m_1) = c_1 \end{aligned}$$



The adversary can check if it guessed correctly!

Chained CBC is not CPA-Secure!

Suppose an adversary is given (c_0, c_1, c_2, c_3) and wants to tell if m_1 was m_1^a or m_1^b

Ad

If n

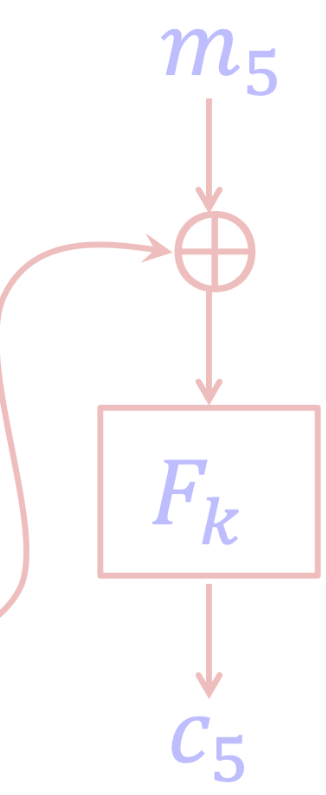
Moral of the story?

Seemingly benign changes can have huge security implications!

Someone just wanted to save on bandwidth by not sending over a new IV each time :(

Use things only as intended unless you can prove security of the modified version (based on reliable assumptions)

$= F_k(IV \oplus m_1) = c_1$

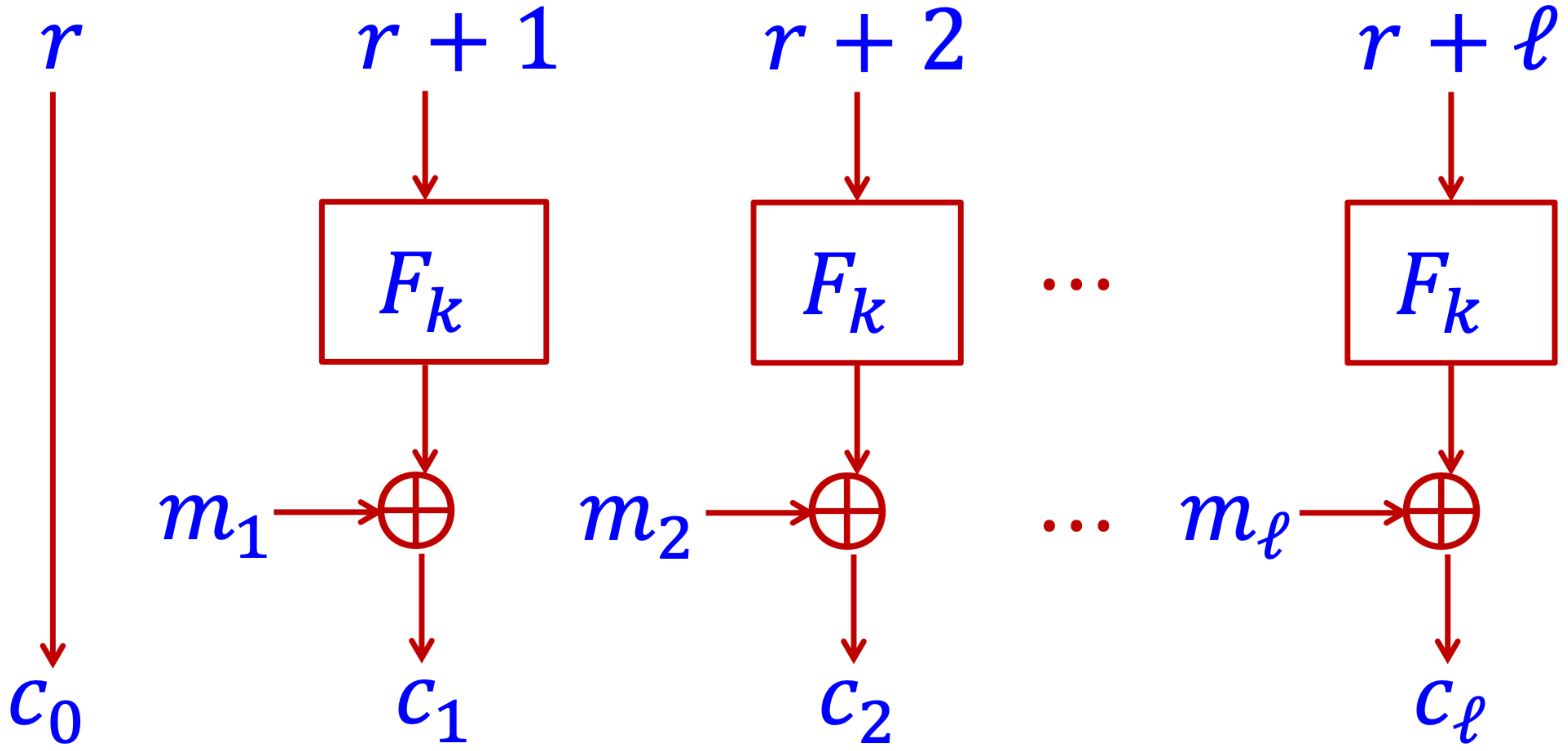


The adversary can check if it guessed correctly!

Counter (CTR) Mode

Our randomly sampled IV (r) will be used as the input to a PRF and incremented for each part of the message

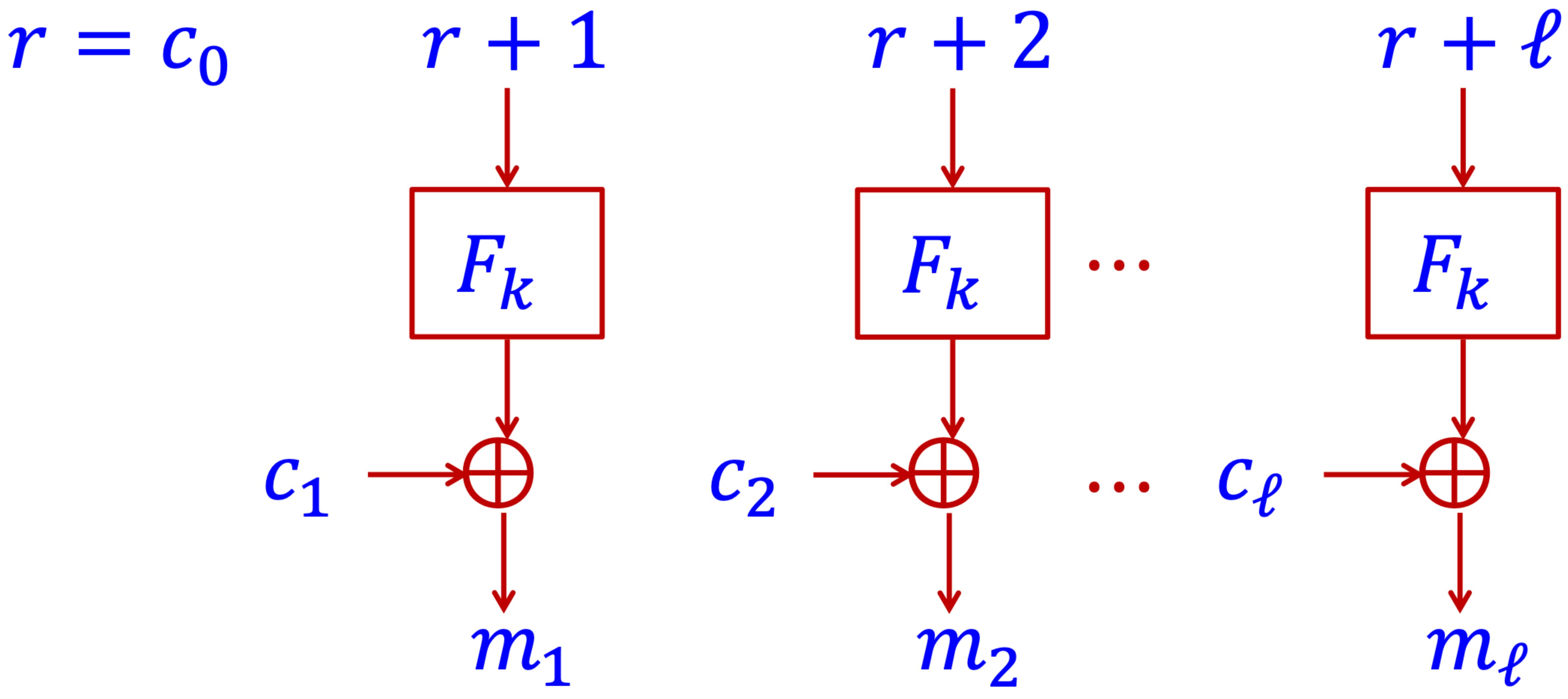
$$
\begin{aligned}
c &= (c_0, c_1, c_2, \ldots, c_\ell) \\
&= (r, F_k(r+1) \oplus m_1, F_k(r+2) \oplus m_2, \ldots, F_k(r+\ell) \oplus m_\ell)
\end{aligned}
$$



Counter (CTR) Mode

To decrypt, you input the cipher texts into the PRF again:

$$\mathsf{Dec}_k(c_0, c_1, c_2, \ldots, c_\ell)$$
$$= F_k(c_0 + 1) \oplus c_1, F_k(c_0 + 2) \oplus c_2, \ldots, F_k(c_0 + \ell) \oplus c_\ell)$$



CTR Properties

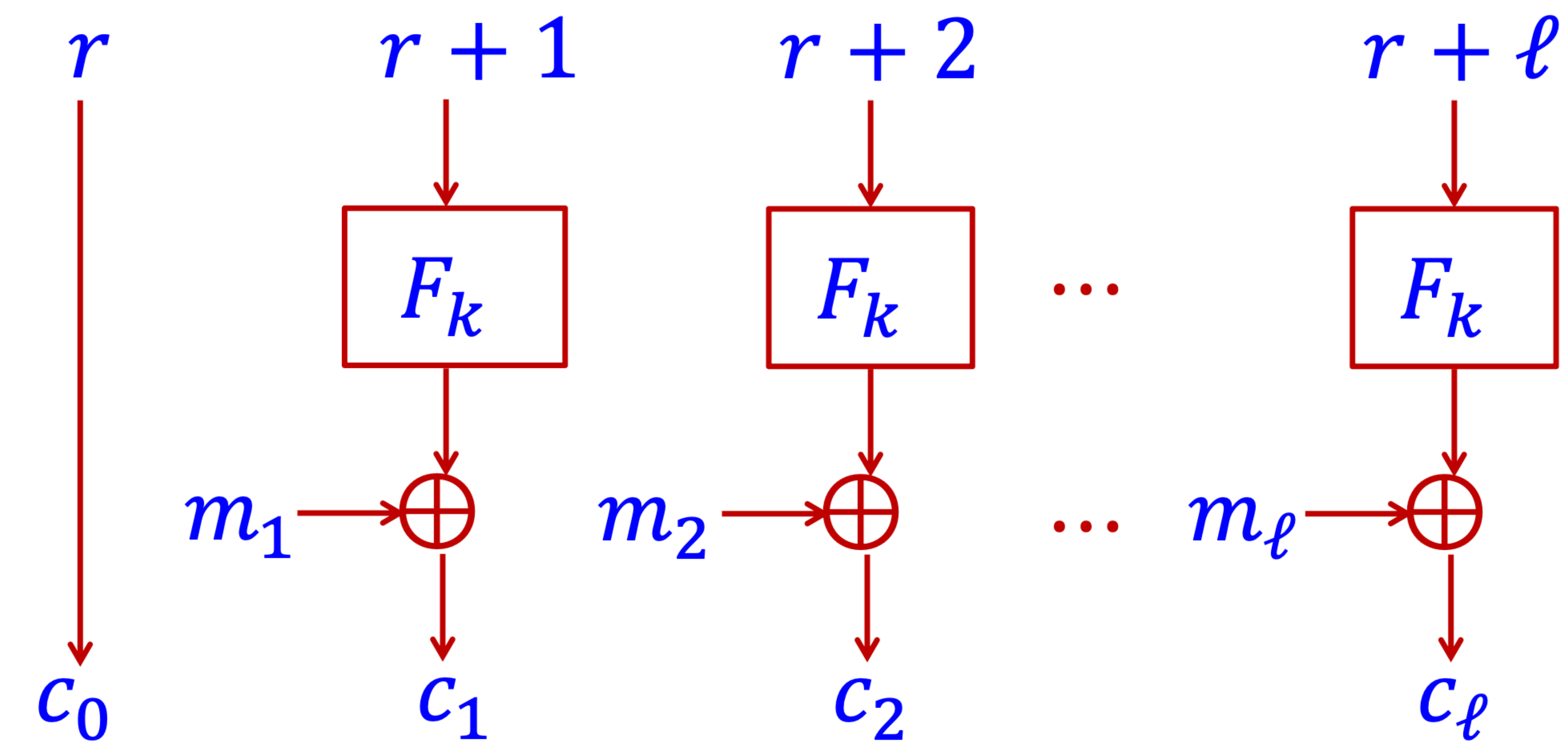
- **Theorem**: If F is a PRF, then CTR is CPA-secure

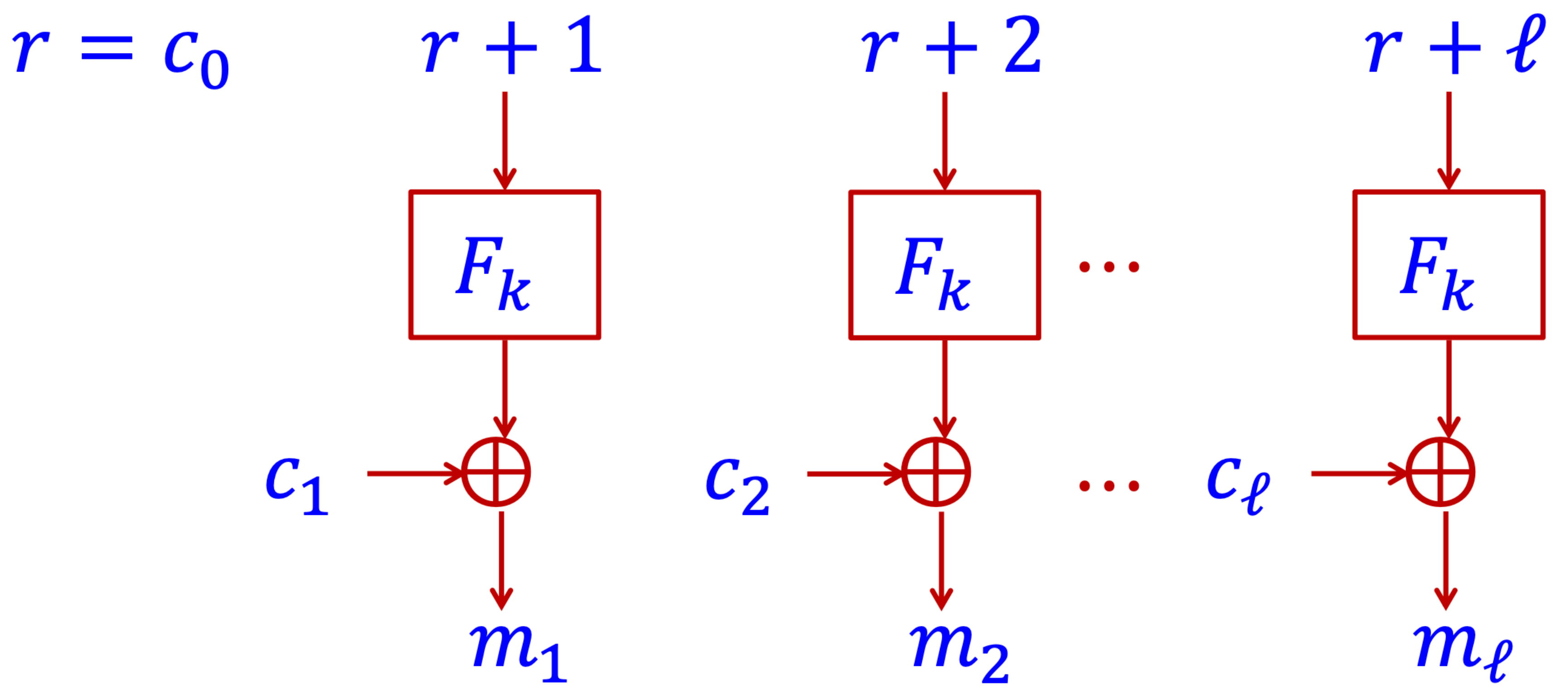
- Expansion is only one block

- Encryption and decryption can be done in **parallel**

- Notice we don't need to invert to decrypt

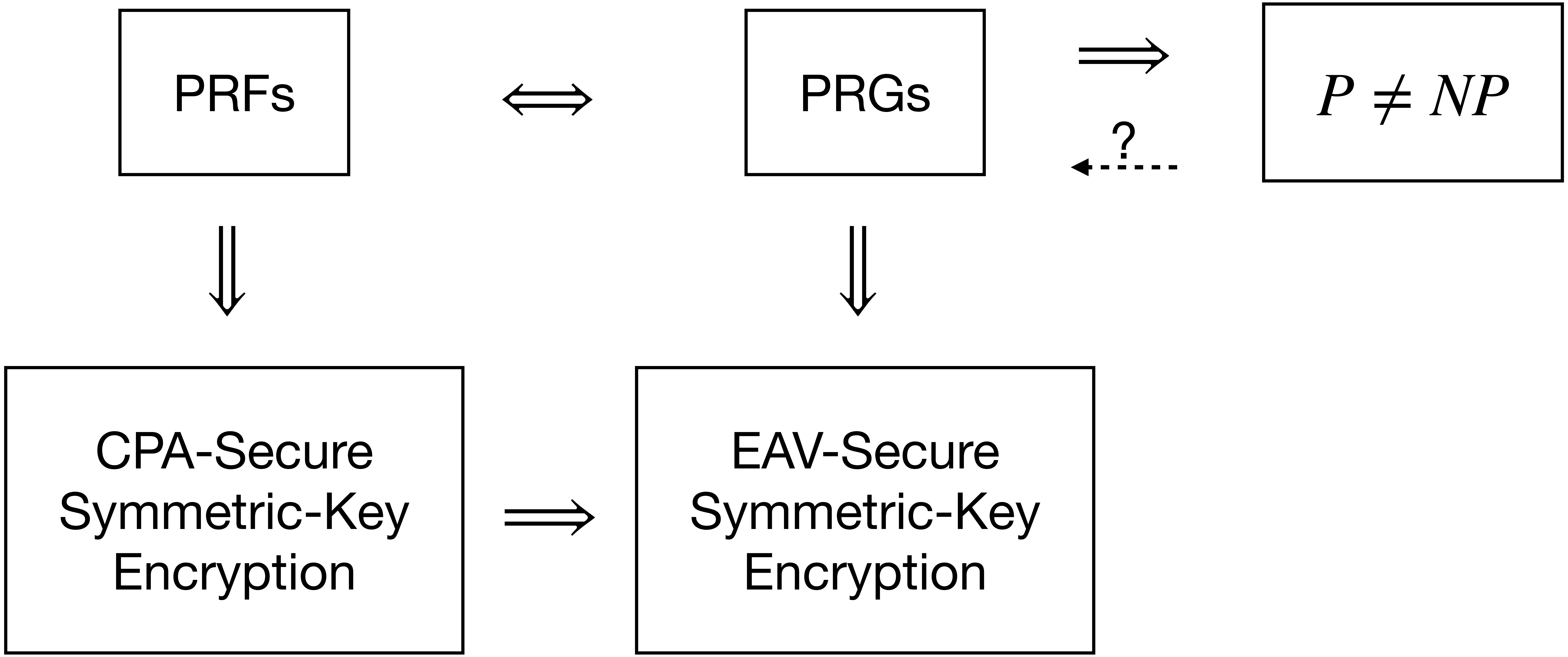
- We also don't need padding! We can truncate





Back to theoretical constructions

The World of Crypto Primitives We’ve Seen So Far

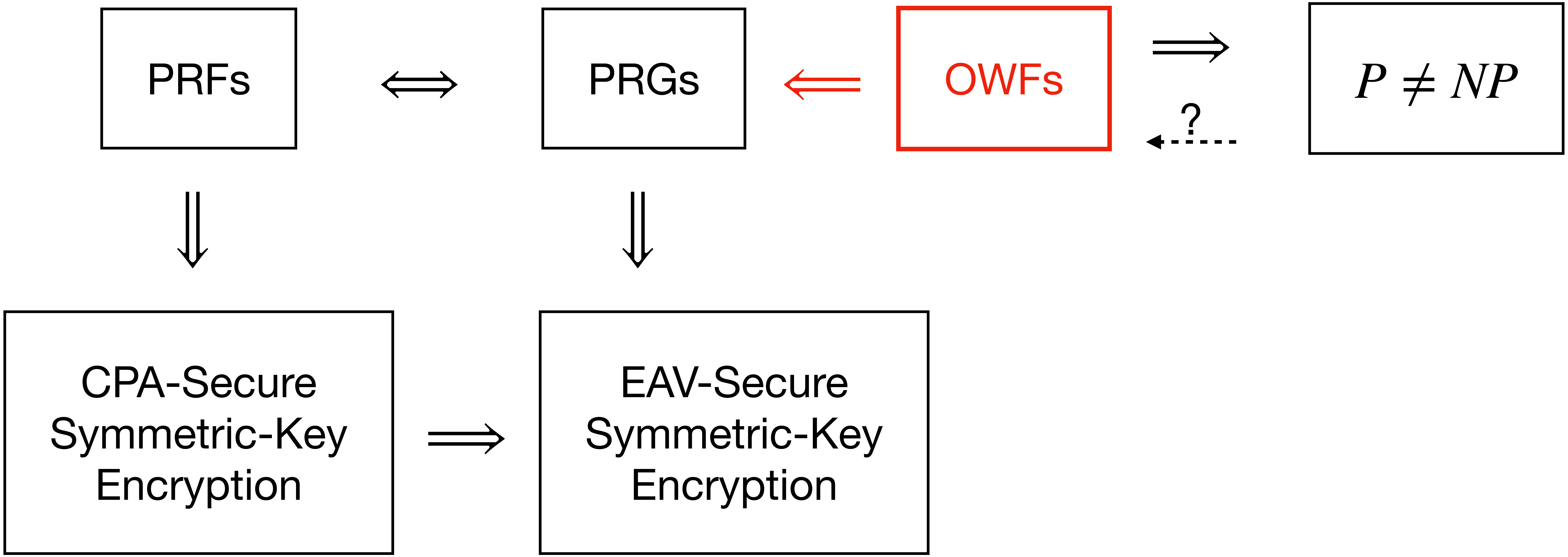


PRFs $\Longleftrightarrow$ PRGs $\Longrightarrow$ $P \neq NP$

?

CPA-Secure Symmetric-Key Encryption $\Longrightarrow$ EAV-Secure Symmetric-Key Encryption

The World of Crypto Primitives We’ve Seen So Far



PRFs

PRGs

OWFs

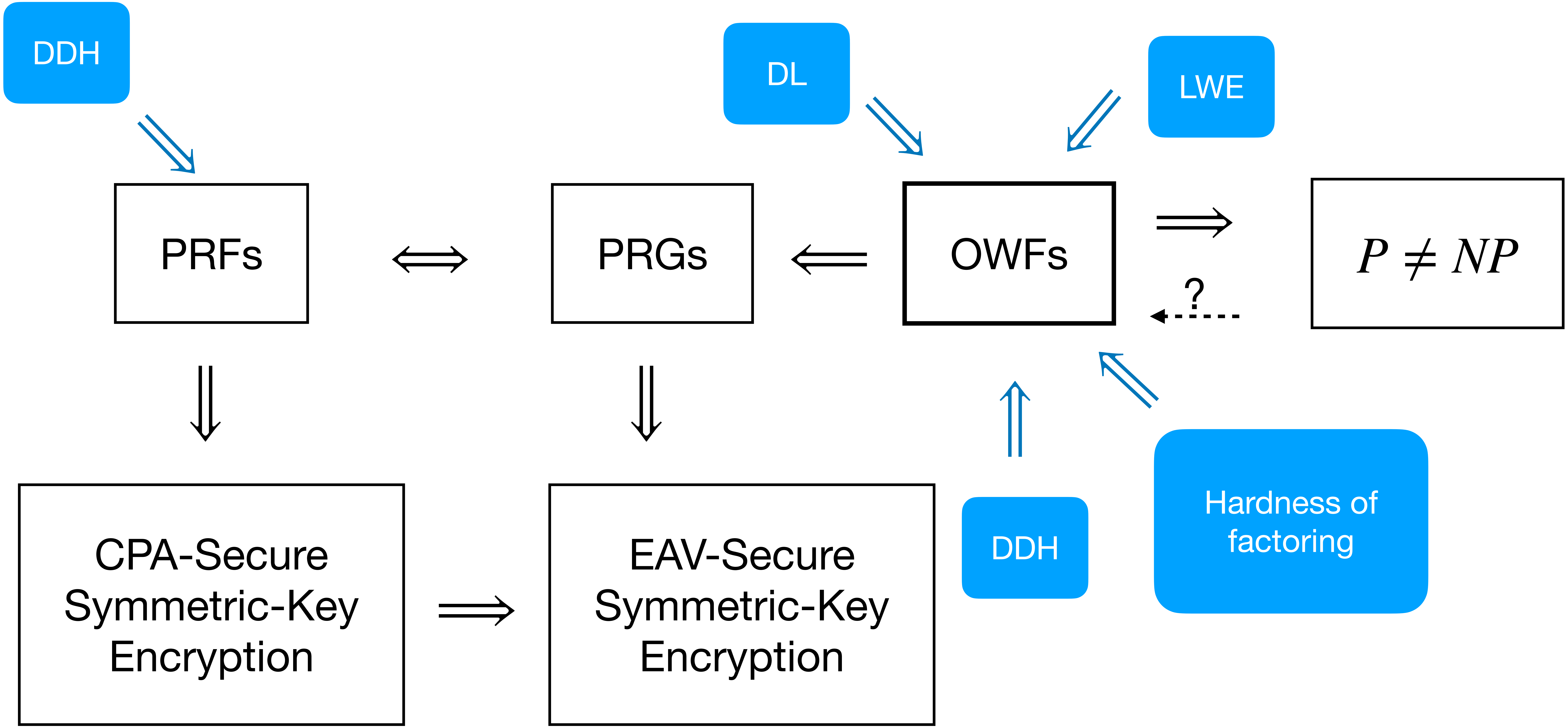
$P \neq NP$

?

CPA-Secure Symmetric-Key Encryption

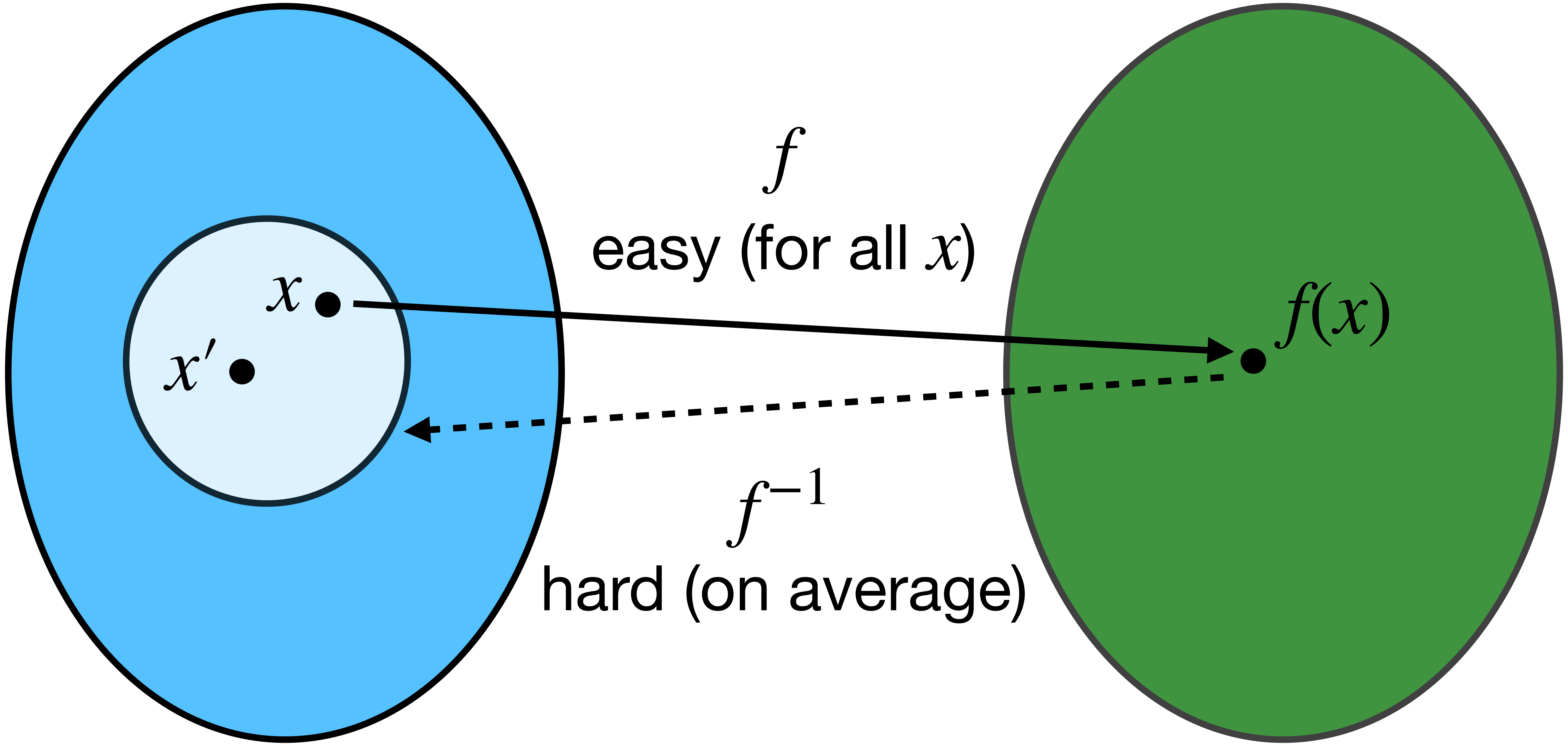
EAV-Secure Symmetric-Key Encryption

Later: Constructions from Assumptions



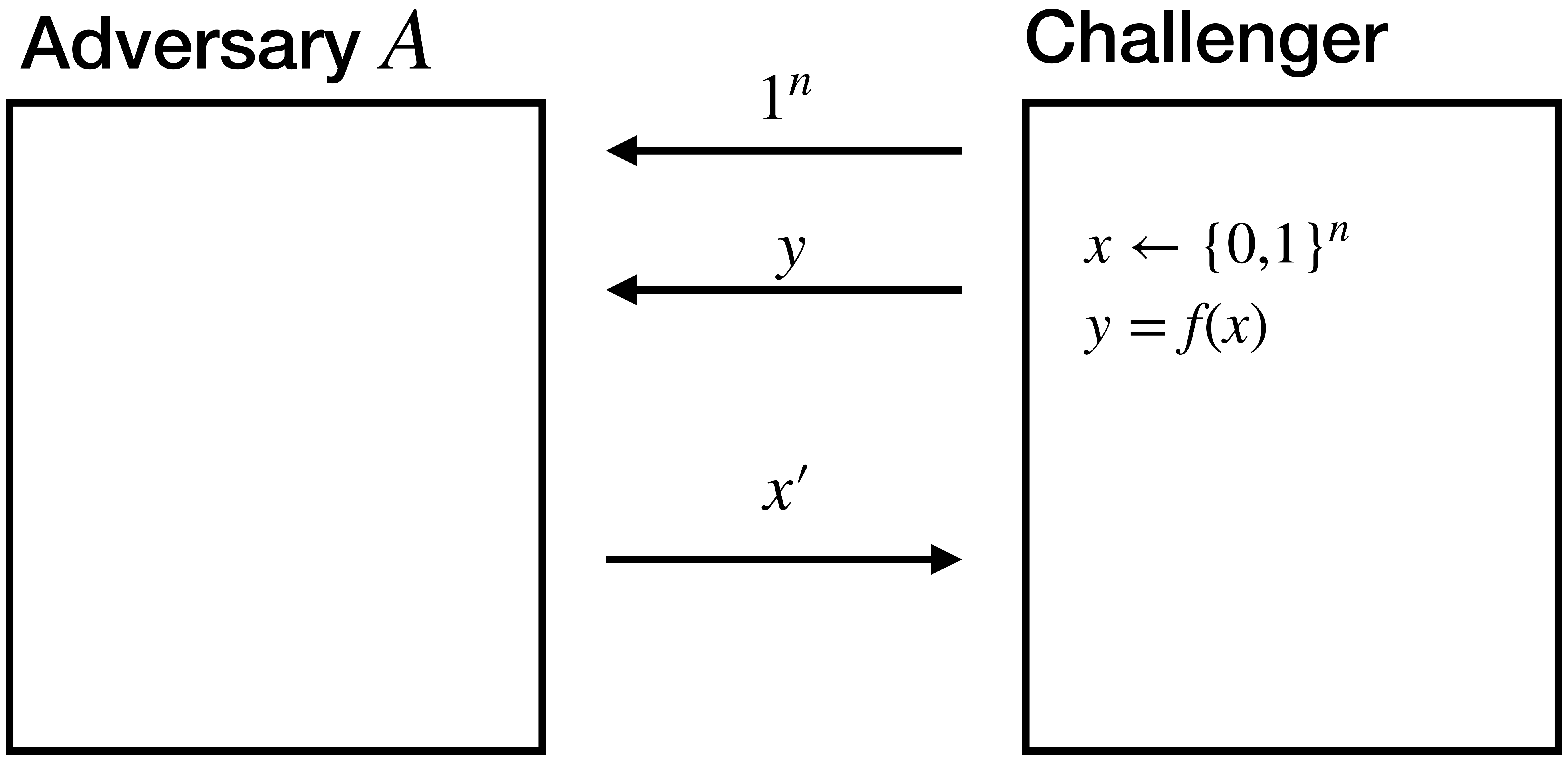
One-Way Functions (OWFs)

A function $f: \{0,1\}^* \to \{0,1\}^*$ that is easy to compute but hard to invert



One-Way Functions

Given $f : \{0,1\}^* \to \{0,1\}^*$ and an adversary A, consider the experiment $\mathtt{Invert}_{f,A}(n)$:



$\mathtt{Invert}_{f,A}(n) = 1$ if $f(x') = y$
and 0 otherwise

One-Way Functions

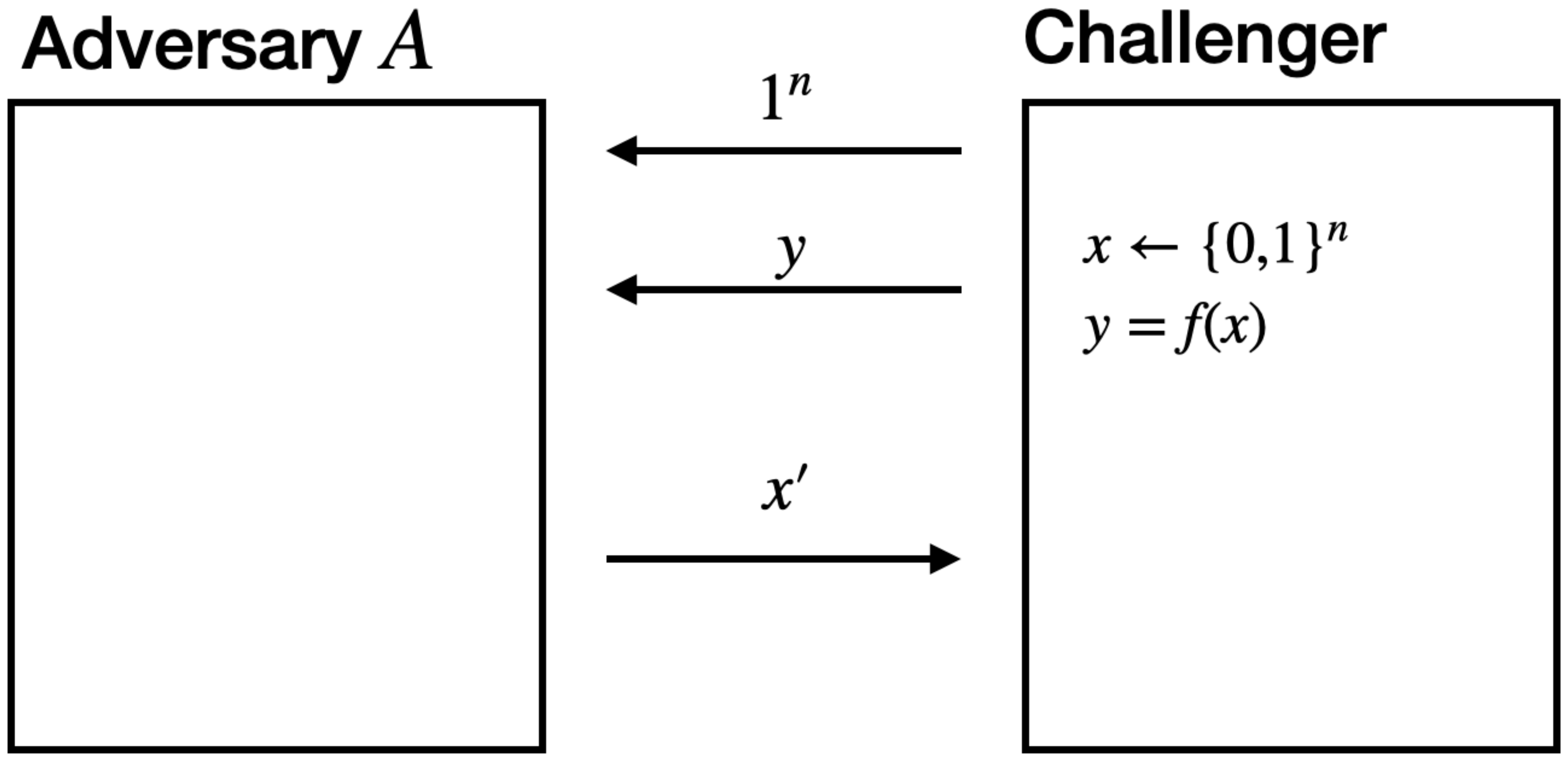
Given $f : \{0,1\}^* \rightarrow \{0,1\}^*$ and an adversary A, consider the experiment $\mathsf{Invert}_{f,A}(n)$:

Definition:

$f : \{0,1\}^* \rightarrow \{0,1\}^*$ is a **one-way function** if f can be computed in polynomial time, and for every PPT adversary A there exists a negligible function $\epsilon(\cdot)$ such that

$$\Pr[\mathsf{Invert}_{f,A}(n) = 1] \leq \epsilon(n)$$

where the probability is taken over the random coins used by A and by the experiment.



Adversary A

Challenger

1^n

y

x'

$x \leftarrow \{0,1\}^n$
$y = f(x)$

$\mathsf{Invert}_{f,A}(n) = 1$ if $f(x') = y$
and 0 otherwise

One-Way Functions

Given $f: \{0,1\}^* \rightarrow \{0,1\}^*$ and an ad der the experiment $\mathsf{Invert}_{f,A}(n)$:

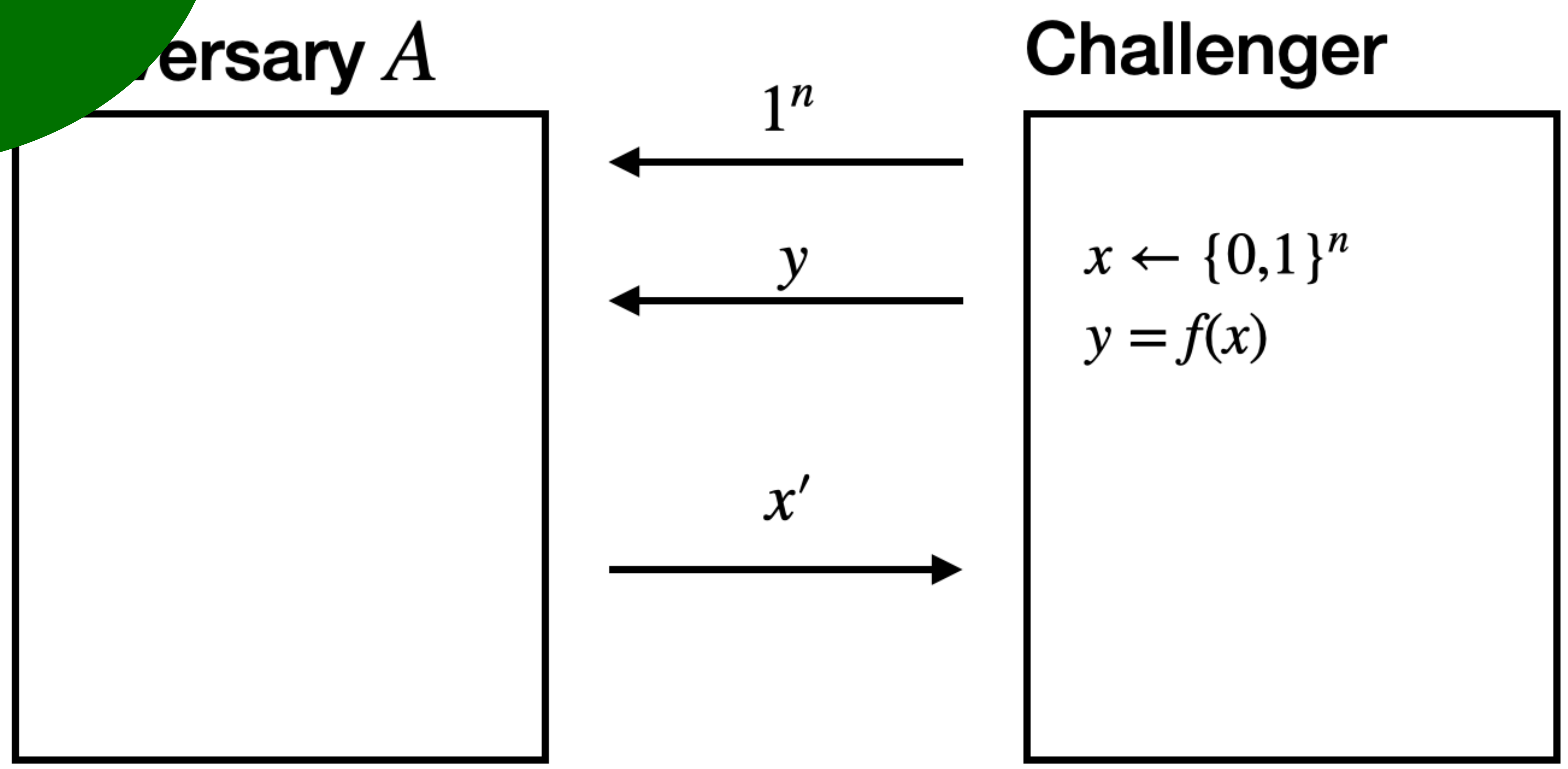
f is efficiently computable

Definition:

$f: \{0,1\}^* \rightarrow \{0,1\}^*$ is a **one-way fun on** if f can be computed in polynomial time, and for every PPT adversary A there exists a negligible function $\epsilon(\cdot)$ such that

$$\Pr[\mathsf{Invert}_{f,A}(n) = 1] \leq \epsilon(n)$$

where the probability is taken over the random coins used by A and by the experiment.



$\mathsf{Invert}_{f,A}(n) = 1$ if $f(x') = y$
and 0 otherwise

One-Way Functions

Given $f : \{0,1\}^* \to \{0,1\}^*$ and an ad der the experiment $\mathsf{Invert}_{f,A}(n)$:

f is efficiently computable

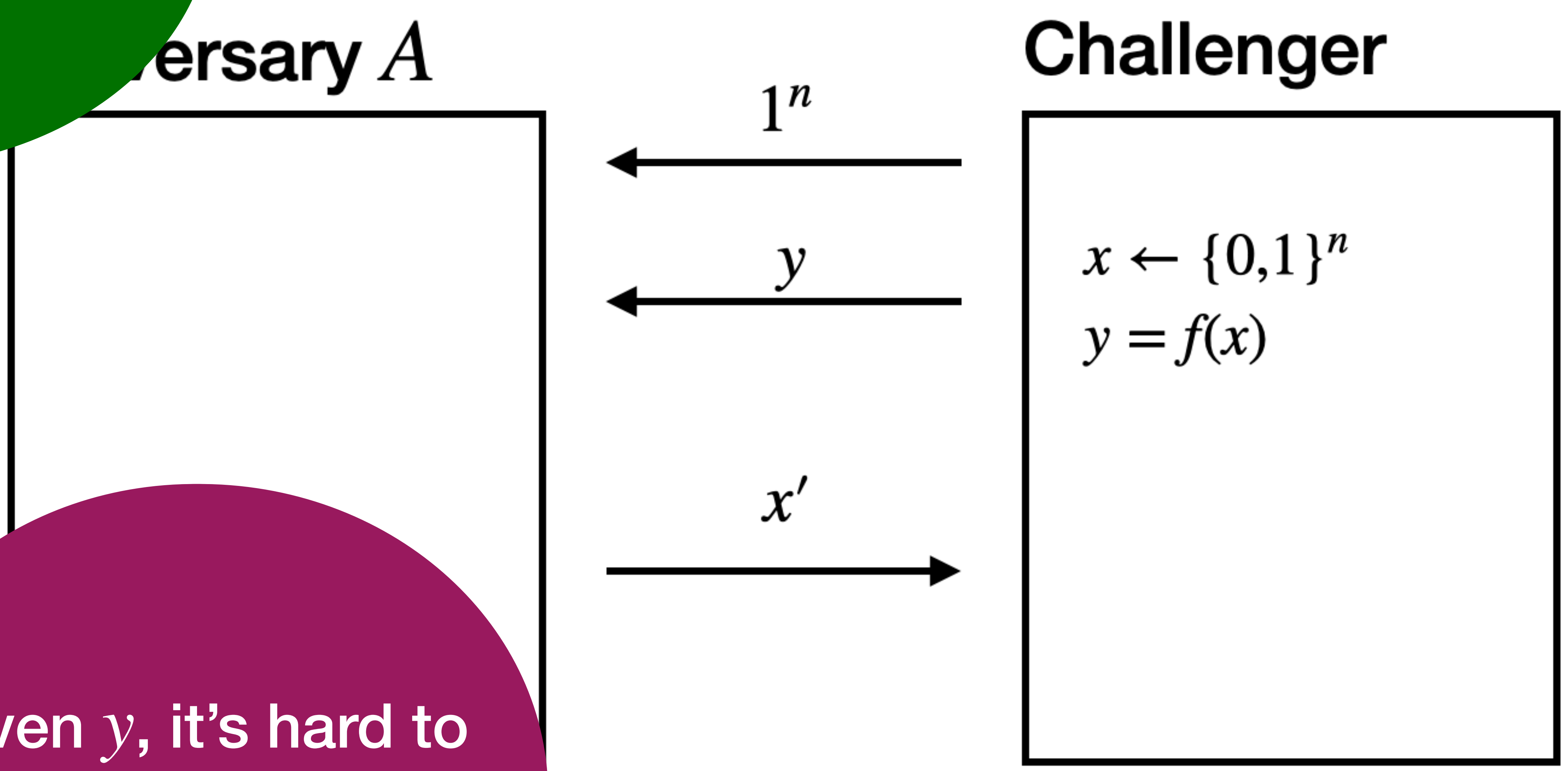
Definition:

$f : \{0,1\}^* \to \{0,1\}^*$ is a **one-way fun ion** if f can be computed in polynomial time, and for every PPT adversary A there exists a negligible function $\epsilon(\cdot)$ such that

$$\Pr[\mathsf{Invert}_{f,A}(n) = 1] \leq \epsilon(n)$$

where the probability is taken over the random coins used by A and by the experiment.

Given y, it's hard to find an input that produces it



$\mathsf{Invert}_{f,A}(n) = 1$ if $f(x') = y$
and 0 otherwise

Some facts about OWFs

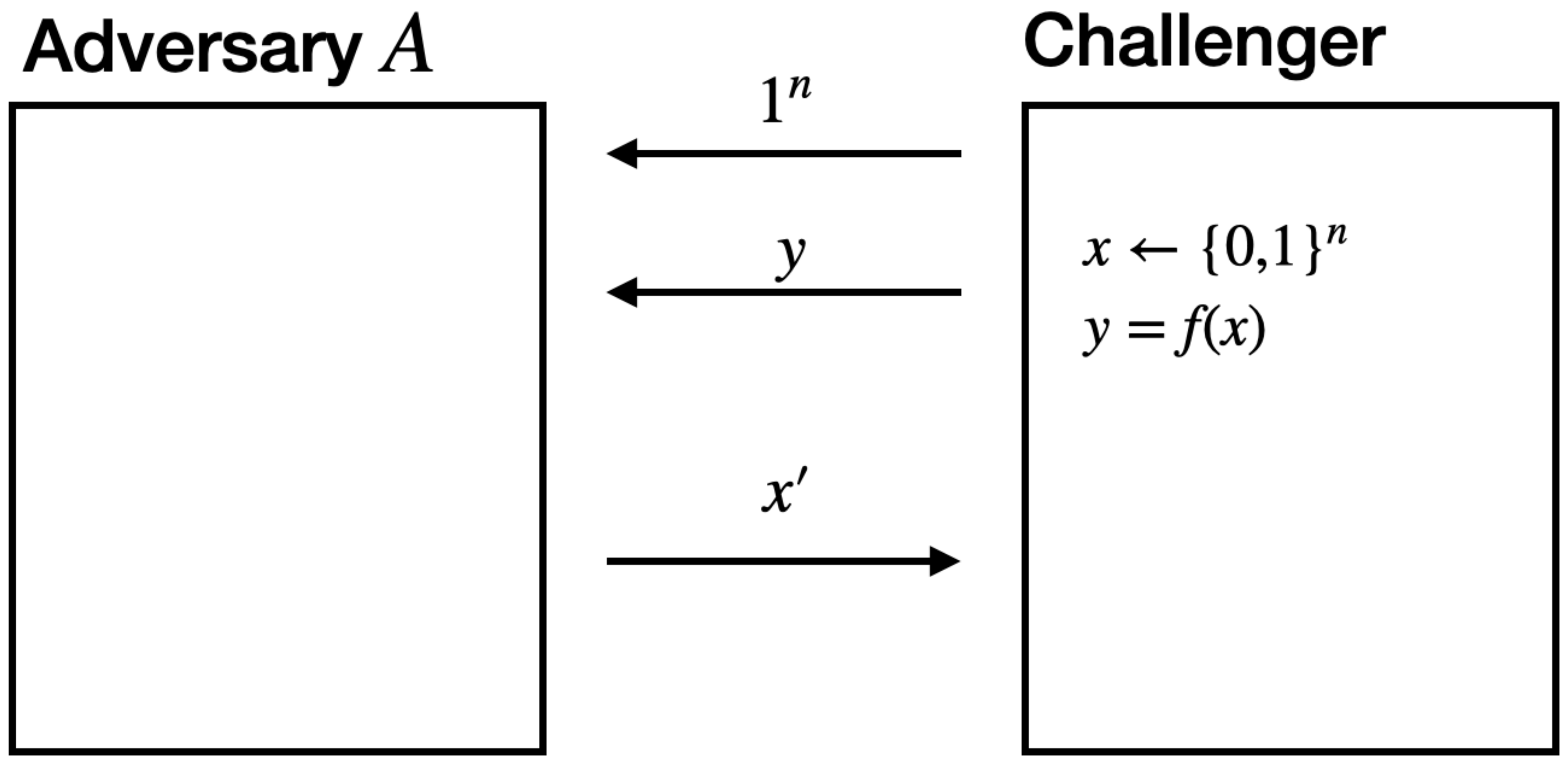
- A OWF does not need to hide all of its input!

- A OWF des not guarantee hardness of inverting *every* input

- If OWF exist, then $P \neq NP$

- OWF implies PRG

- If f is a OWF and a bijection, then f is a one-way permutation (OWP)



$\text{Invert}_{f,A}(n) = 1$ if $f(x') = y$
and 0 otherwise

Next Time

- OWFs/OWPs continued
- CCA-Security
- MACs