



Odds and ends

Key Derivation

Deriving many keys from one

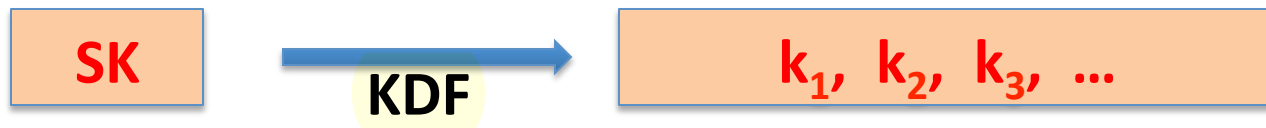
Typical scenario. a single source key (SK) is sampled from:

- Hardware random number generator
- A key exchange protocol (discussed later)

Need many keys to secure session:

- unidirectional keys; multiple keys for nonce-based CBC.

Goal: generate many keys from this one source key



When source key is uniform

F: a PRF with key space K and outputs in $\{0,1\}^n$

Suppose source key SK is uniform in K

- Define Key Derivation Function (KDF) as:

KDF(SK , CTX , L) :=

$F(SK, (CTX \parallel 0)) \parallel F(SK, (CTX \parallel 1)) \parallel \cdots \parallel F(SK, (CTX \parallel L))$

CTX: a string that uniquely identifies the application

KDF(SK, CTX, L) :=

$F(\text{SK}, (\text{CTX} \parallel 0)) \parallel F(\text{SK}, (\text{CTX} \parallel 1)) \parallel \dots \parallel F(\text{SK}, (\text{CTX} \parallel L))$

What is the purpose of CTX?

- ⇒
- ☐ Even if two apps sample same SK they get indep. keys
 - ☐ It's good practice to label strings with the app. name
 - ☐ It serves no purpose
 - ☐

What if source key is not uniform?

Recall: PRFs are pseudo random only when key is uniform in K

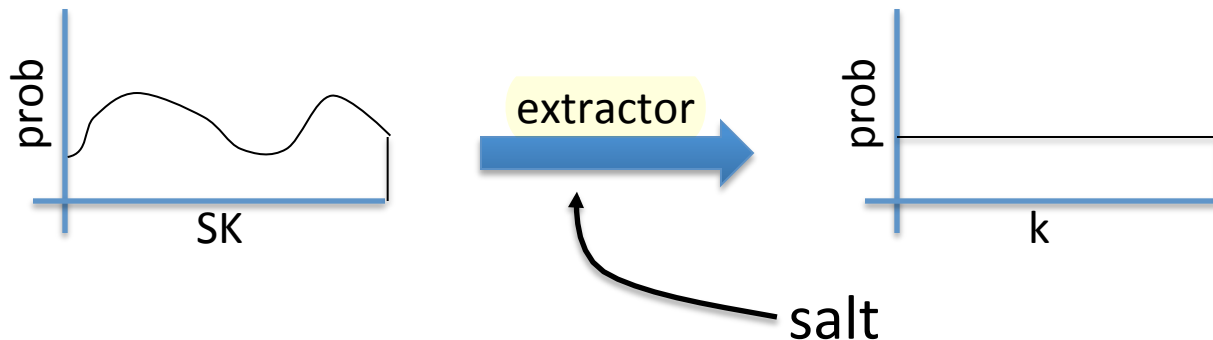
- SK not uniform \Rightarrow PRF output may not look random

Source key often not uniformly random:

- Key exchange protocol: key uniform in some subset of K
- Hardware RNG: may produce biased output

Extract-then-Expand paradigm

Step 1: **extract** pseudo-random key k from source key SK



salt: a fixed non-secret string chosen at random

step 2: **expand** k by using it as a PRF key as before

HKDF: a KDF from HMAC

Implements the extract-then-expand paradigm:

- extract: use $k \leftarrow \text{HMAC}(\text{salt}, \text{SK})$
- Then expand using HMAC as a PRF with key k

Password-Based KDF (PBKDF)

Deriving keys from passwords:

- Do not use HKDF: passwords have insufficient entropy
- Derived keys will be vulnerable to dictionary attacks (more on this later)

PBKDF defenses: **salt** and a **slow hash function**

Standard approach: **PKCS#5** (PBKDF1)

$H^{(c)}(\text{pwd} \parallel \text{salt})$: iterate hash function c times

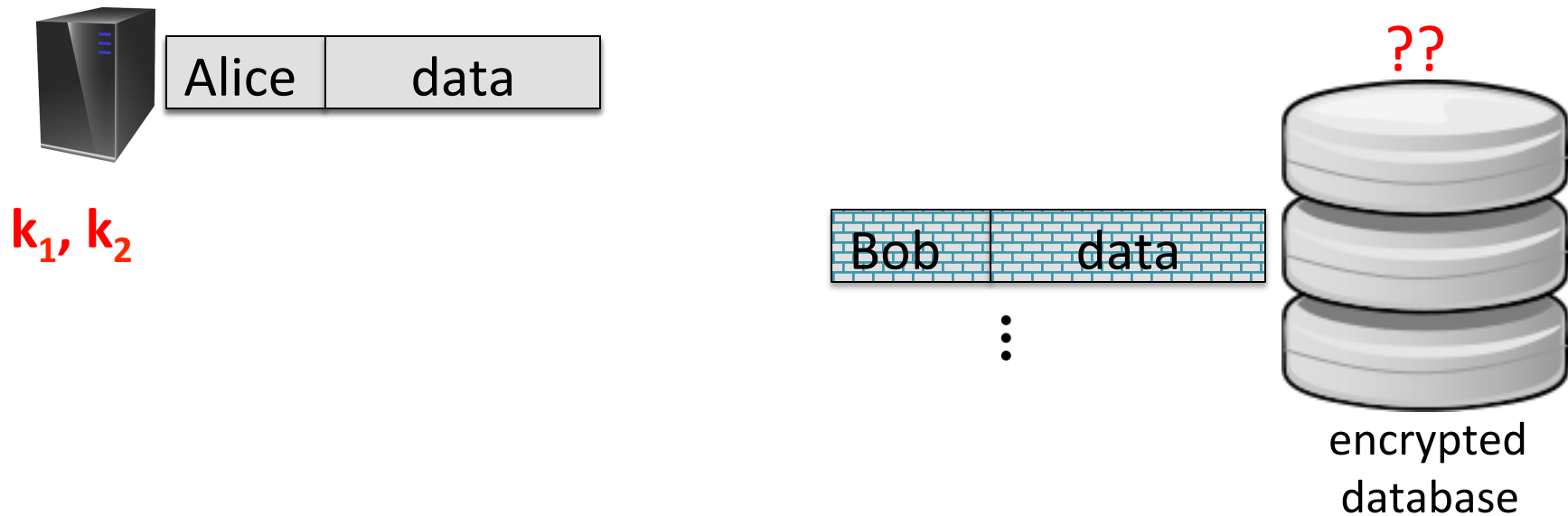
End of Segment



Odds and ends

Deterministic Encryption

The need for det. Encryption (no nonce)



The need for det. Encryption (no nonce)



k_1, k_2

Alice	data
Bob	data

⋮



encrypted
database

Later:



Retrieve record $E(k_1, \text{"Alice"})$

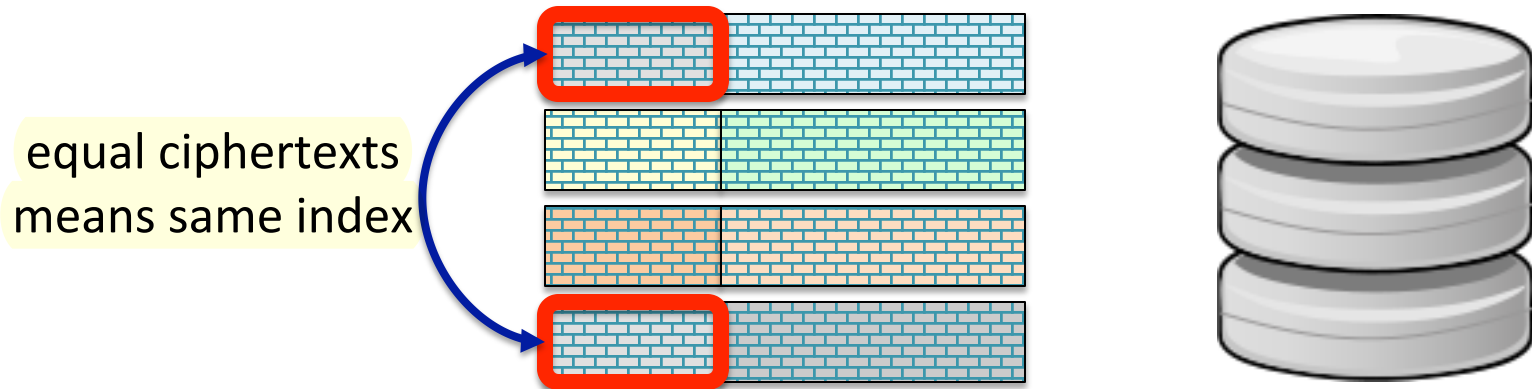
Alice	data
-------	------

det. enc. enables later lookup

Problem: det. enc. cannot be CPA secure

The problem: attacker can tell when two ciphertexts encrypt the same message \Rightarrow leaks information

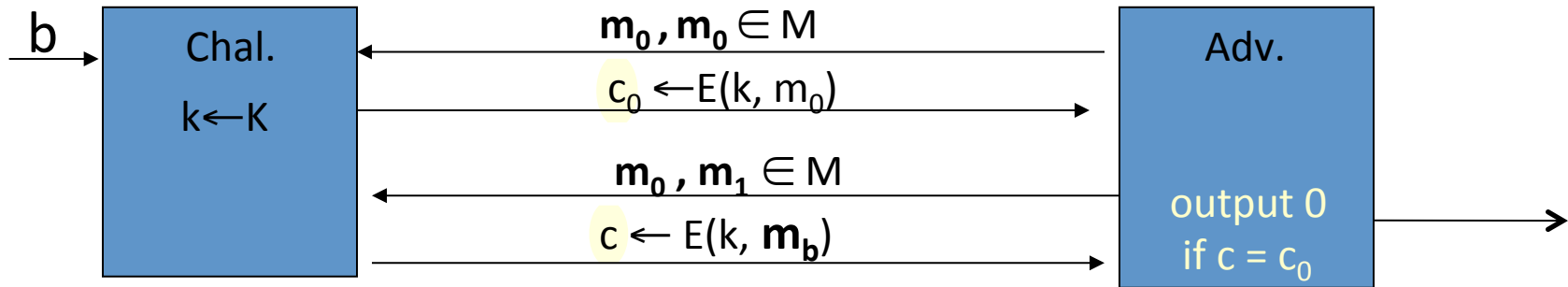
Leads to significant attacks when message space M is small.



Problem: det. enc. cannot be CPA secure

The problem: attacker can tell when two ciphertexts encrypt the same message \Rightarrow leaks information

Attacker wins CPA game:



A solution: the case of unique messages

Suppose encryptor never encrypts same message twice:

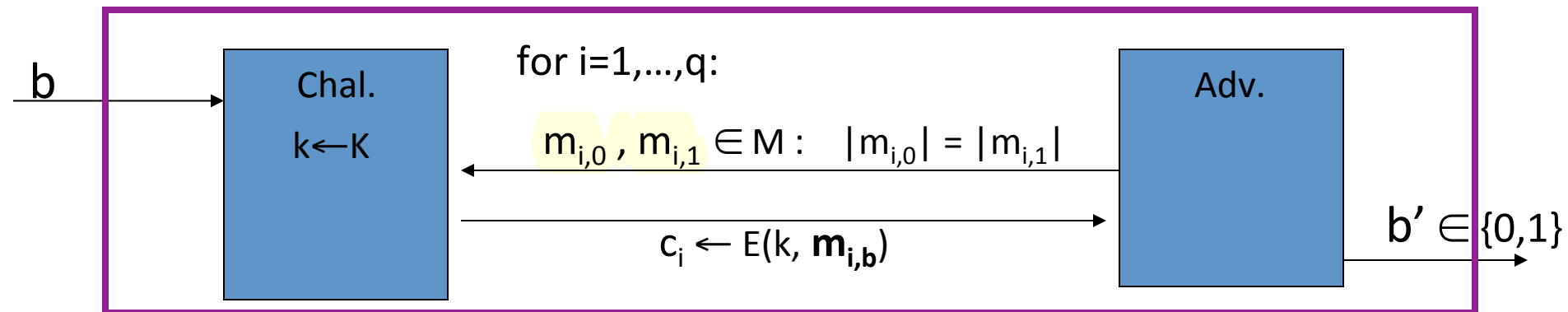
the pair (k, m) never repeats

This happens when encryptor:

- Chooses messages at random from a large msg space (e.g. keys)
- Message structure ensures uniqueness (e.g. unique user ID)

Deterministic CPA security

$E = (E, D)$ a cipher defined over (K, M, C) . For $b=0,1$ define $\text{EXP}(b)$ as:



where $m_{1,0}, \dots, m_{q,0}$ are distinct and $m_{1,1}, \dots, m_{q,1}$ are distinct

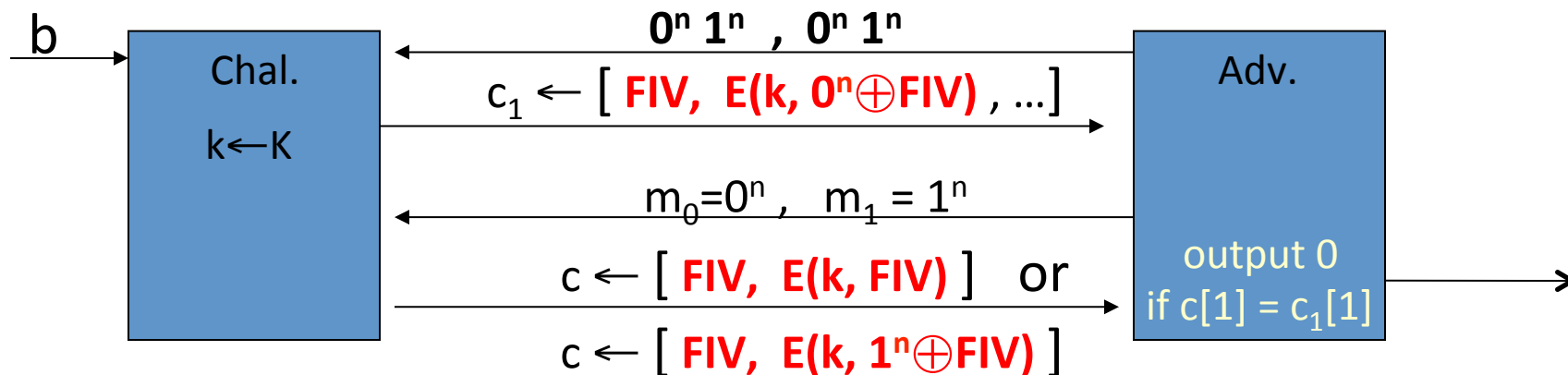
Def: E is **sem. sec. under det. CPA** if for all efficient A :

$$\text{Adv}_{\text{dCPA}}[A, E] = \left| \Pr[\text{EXP}(0)=1] - \Pr[\text{EXP}(1)=1] \right| \text{ is negligible.}$$

A Common Mistake

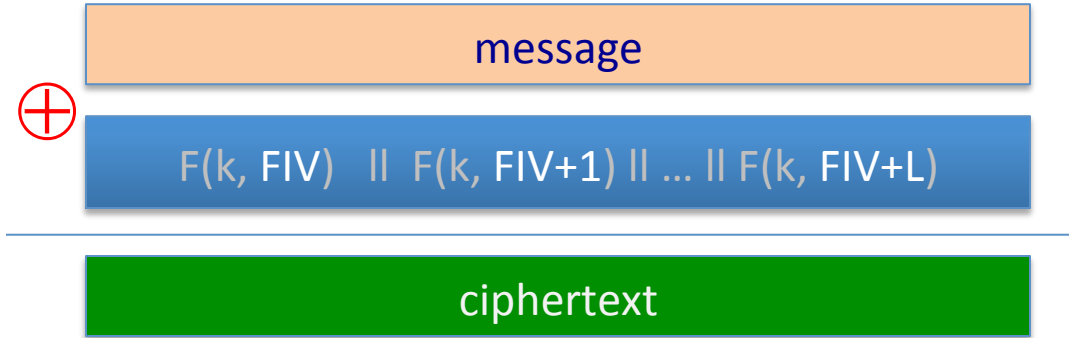
CBC with fixed IV is not det. CPA secure.

Let $E: K \times \{0,1\}^n \rightarrow \{0,1\}^n$ be a secure PRP used in CBC

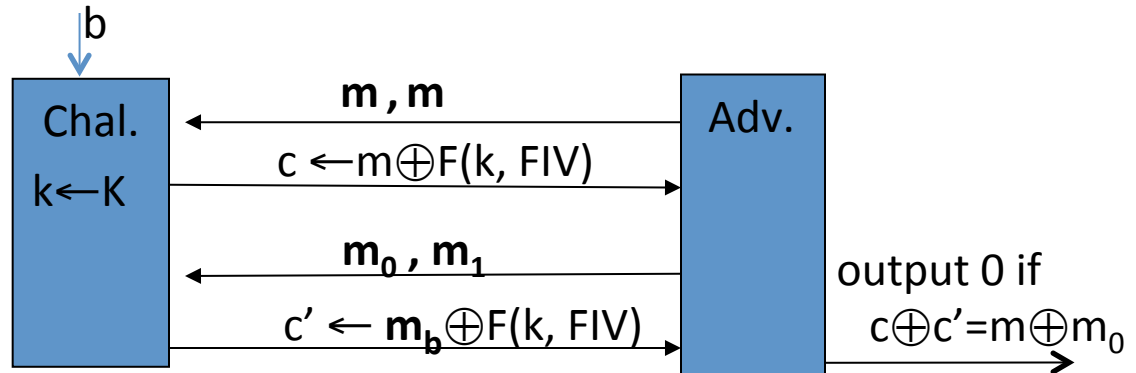


Leads to significant attacks in practice.

Is counter mode with a fixed IV det. CPA secure?



- ☐ Yes
- ☒ No
- ☐ It depends
- ☐



End of Segment



Odds and ends

Deterministic Encryption
Constructions:
SIV and wide PRP

Deterministic encryption

Needed for maintaining an encrypted database index

- Lookup records by encrypted index

Deterministic CPA security:

- Security if never encrypt same message twice using same key:
the pair (key, msg) is unique

Formally: we defined deterministic CPA security game

Construction 1: Synthetic IV (SIV)

Let (E, D) be a CPA-secure encryption. $E(k, m ; r) \rightarrow c$

Let $F: K \times M \rightarrow R$ be a secure PRF

Define: $E_{\text{det}}((k_1, k_2), m) = \begin{cases} r \leftarrow F(k_1, m) \\ c \leftarrow E(k_2, m ; r) \\ \text{output } c \end{cases}$

Thm: E_{det} is sem. sec. under det. CPA.

Proof sketch: distinct msgs. \Rightarrow all r 's are indist. from random

Well suited for messages longer than one AES block (16 bytes)

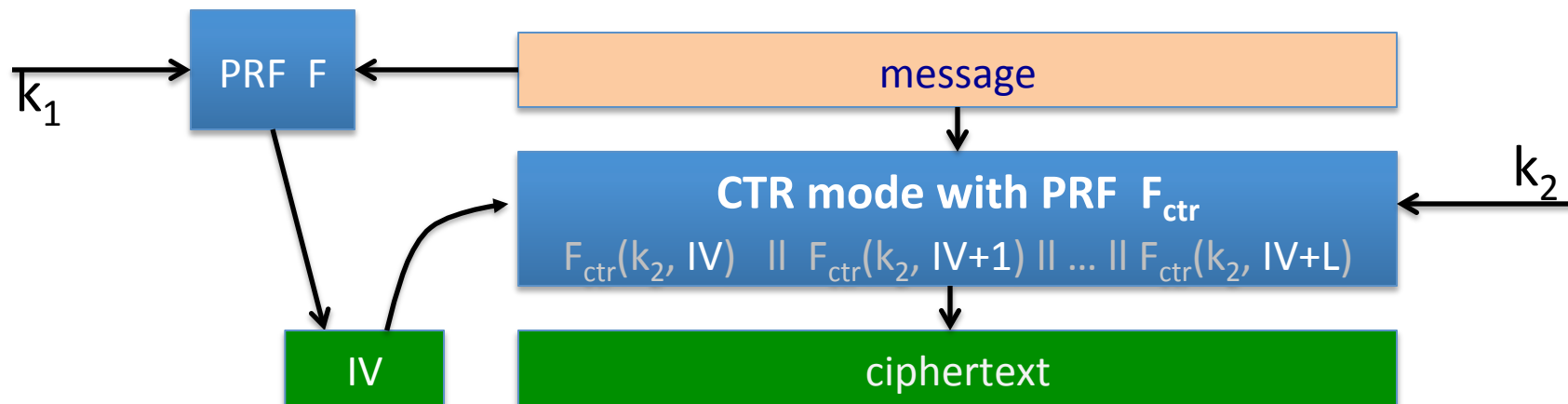
Ensuring ciphertext integrity

Goal: det. CPA security and ciphertext integrity

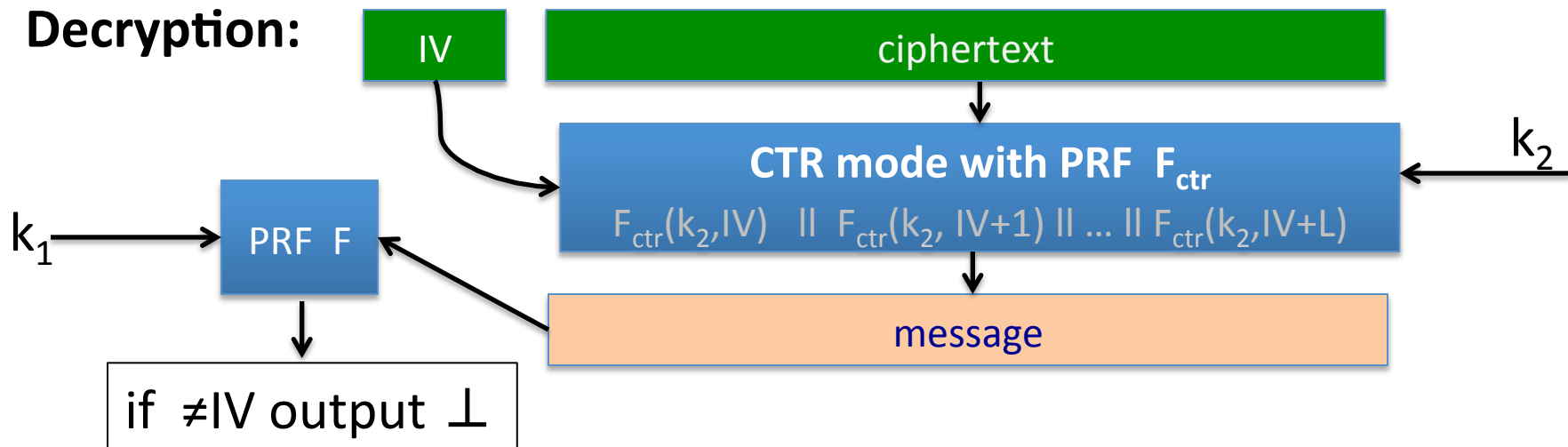
⇒ **DAE: deterministic authenticated encryption**

Consider a SIV special case: SIV-CTR

SIV where cipher is counter mode with rand. IV



Det. Auth. Enc. (DAE) for free



Thm: if F is a secure PRF and CTR from F_{ctr} is CPA-secure
then SIV-CTR from F, F_{ctr} provides DAE

Construction 2: just use a PRP

Let (E, D) be a secure PRP. $E: K \times X \rightarrow X$

Thm: (E, D) is sem. sec. under det. CPA .

Proof sketch: let $f: X \rightarrow X$ be a truly random invertible func.

in $\text{EXP}(0)$ adv. sees: $f(m_{1,0}), \dots, f(m_{q,0})$  q random values in X

in $\text{EXP}(1)$ adv. sees: $f(m_{1,1}), \dots, f(m_{q,1})$

Using AES: Det. CPA secure encryption for 16 byte messages.

Longer messages?? Need PRPs on larger msg spaces ...

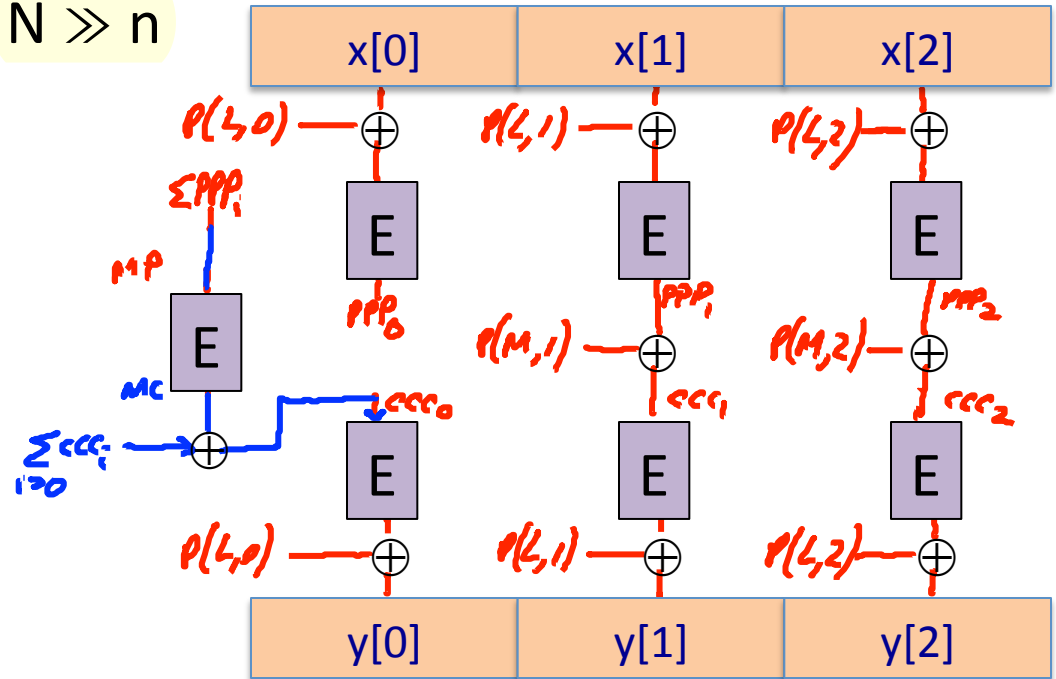
EME: constructing a wide block PRP

Let (E, D) be a secure PRP. $E: K \times \{0,1\}^n \rightarrow \{0,1\}^n$

EME: a PRP on $\{0,1\}^N$ for $N \gg n$

$\text{Key} = (K, L)$

$M \leftarrow MP \oplus MC$



Performance:

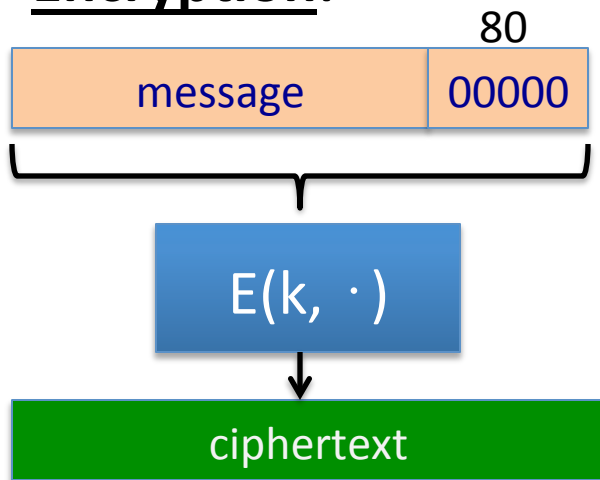
- can be 2x slower than SIV

PRP-based Det. Authenticated Enc.

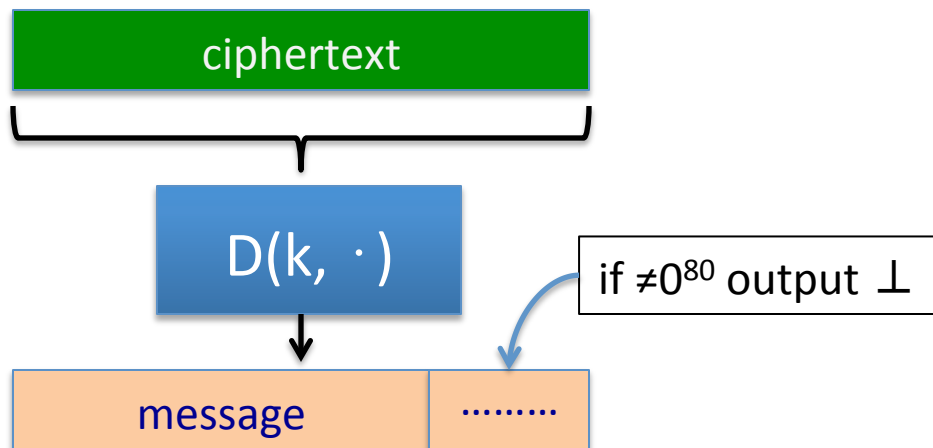
Goal: det. CPA security and ciphertext integrity

⇒ **DAE: deterministic authenticated encryption**

Encryption:



Decryption:

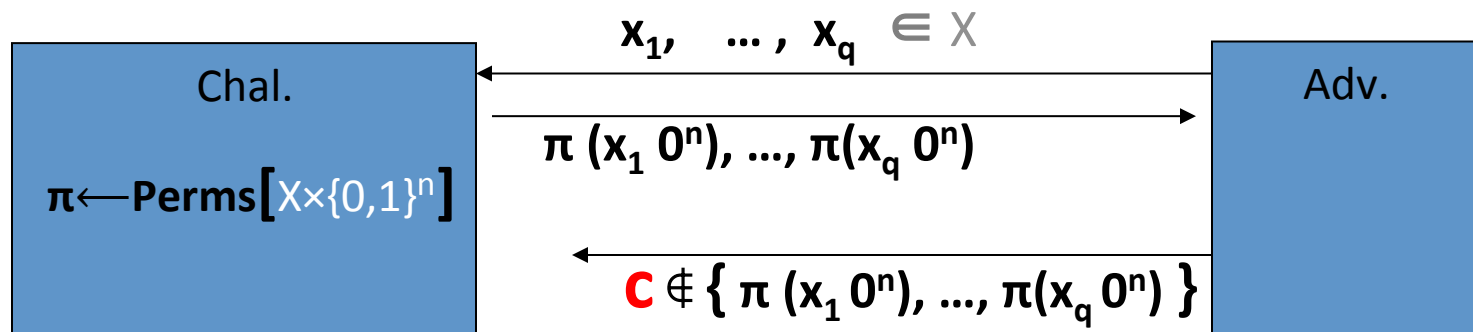


PRP-based Det. Authenticated Enc.

Let (E, D) be a secure PRP. $E: K \times (X \times \{0,1\}^n) \rightarrow X \times \{0,1\}^n$

Thm: $1/2^n$ is negligible \Rightarrow PRP-based enc. provides DAE

Proof sketch: suffices to prove ciphertext integrity



But then $\Pr[\text{LSB}_n(\pi^{-1}(\mathbf{c})) = 0^n] \leq 1/2^n$

End of Segment



Odds and ends

Tweakable encryption

Disk encryption: no expansion

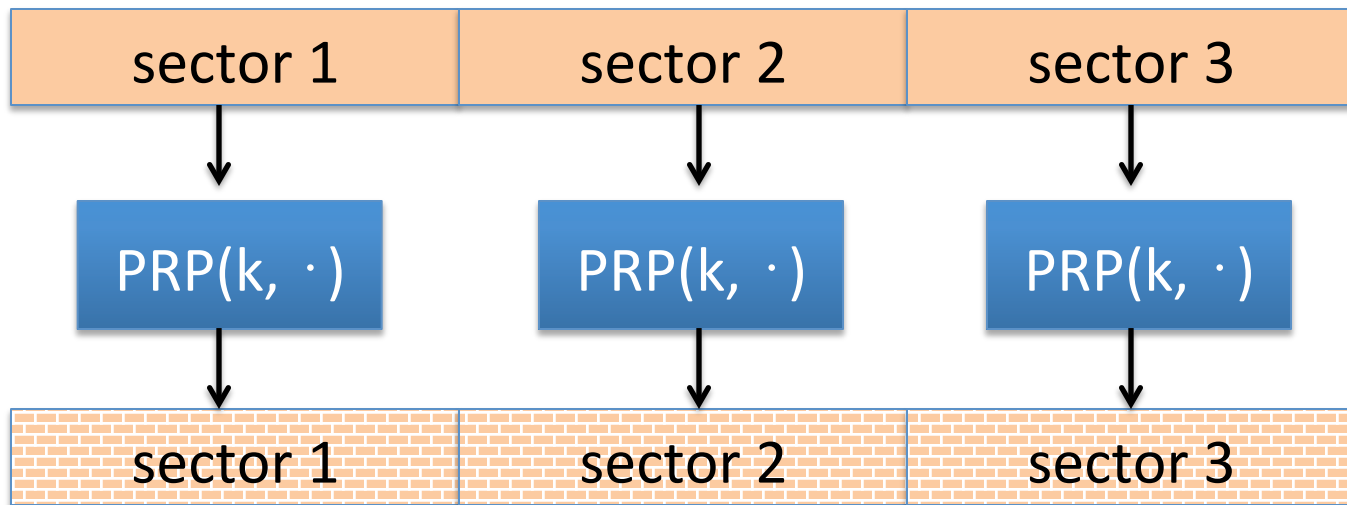
Sectors on disk are fixed size (e.g. 4KB)

⇒ encryption cannot expand plaintext (i.e. $M = C$)

⇒ must use deterministic encryption, no integrity

Lemma: if (E, D) is a det. CPA secure cipher with $M=C$
then (E, D) is a PRP.

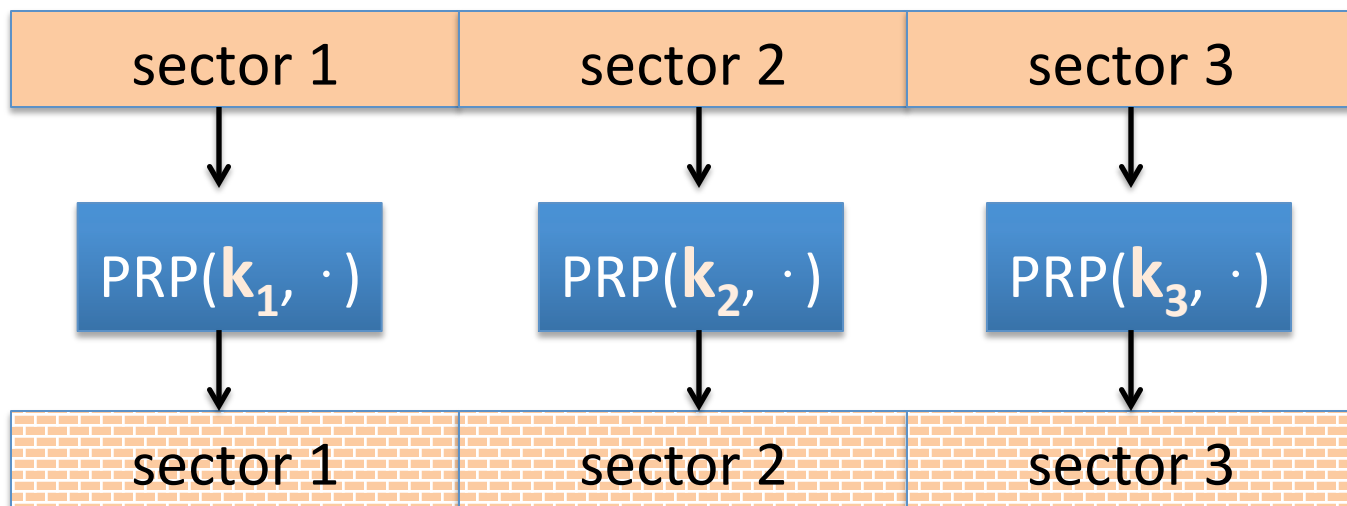
⇒ every sector will need to be encrypted with a PRP



Problem: sector 1 and sector 3 may have same content

- Leaks same information as ECB mode

Can we do better?



Avoids previous leakage problem

- ... but attacker can tell if a sector is changed and then reverted

Managing keys: the trivial construction $k_t = \text{PRF}(k, t)$, $t=1, \dots, L$

Can we do better?

Tweakable block ciphers

Goal: construct many PRPs from a key $k \in K$.

Syntax: $E, D: K \times T \times X \rightarrow X$

for every $t \in T$ and $k \leftarrow K$:

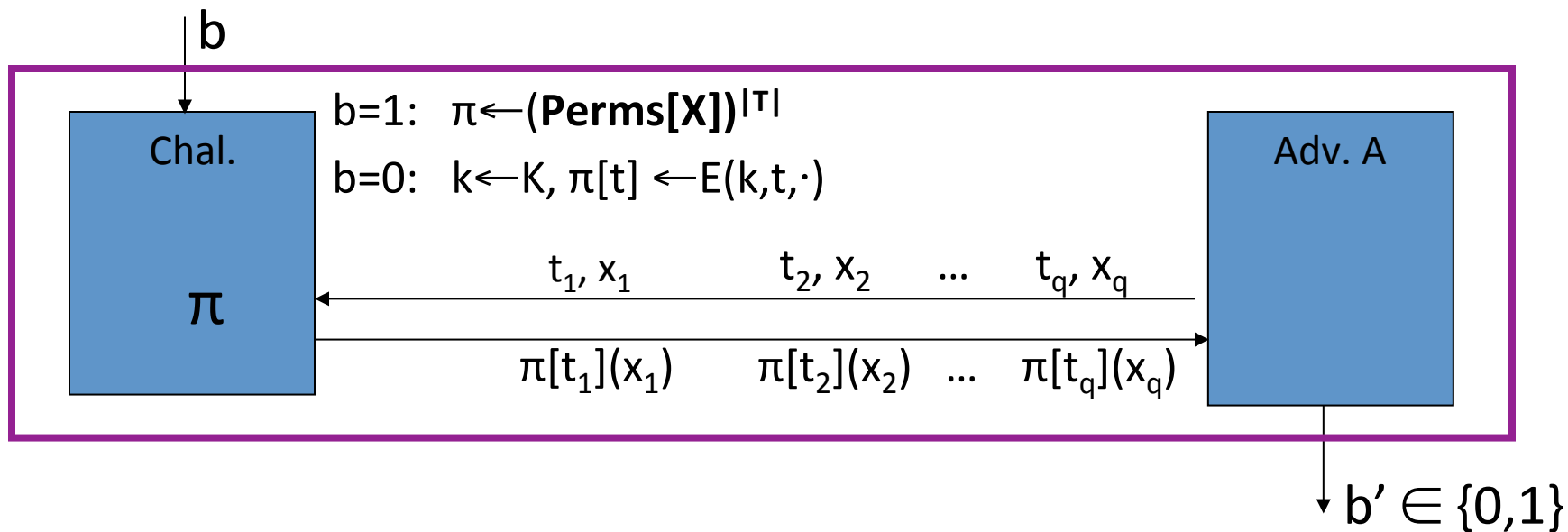
$E(k, t, \cdot)$ is an invertible func. on X , indist. from random

Application: use sector number as the tweak

\Rightarrow every sector gets its own independent PRP

Secure **tweakable** block ciphers

$E, D: K \times T \times X \rightarrow X$. For $b=0,1$ define experiment $\text{EXP}(b)$ as:



- Def: E is a secure tweakable PRP if for all efficient A :

$$\text{Adv}_{\text{tPRP}}[A, E] = \left| \Pr[\text{EXP}(0)=1] - \Pr[\text{EXP}(1)=1] \right| \text{ is negligible.}$$

Example 1: the trivial construction

Let (E,D) be a secure PRP, $E: K \times X \rightarrow X$.

- The trivial tweakable construction: (suppose $K = X$)

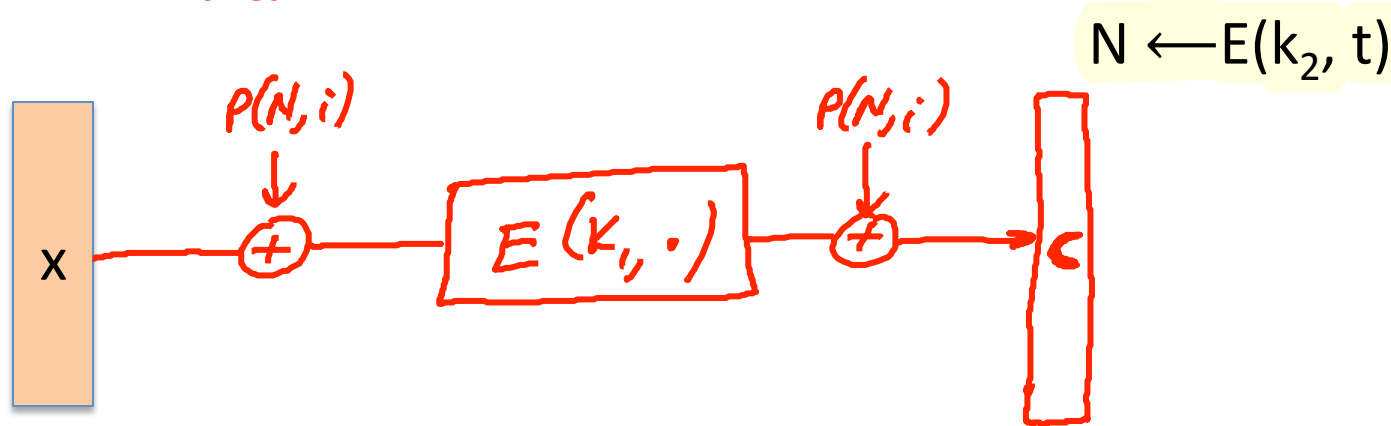
$$E_{\text{tweak}}(k, t, x) = E(E(k, t), x)$$

\Rightarrow to encrypt n blocks need $2n$ evals of $E(.,.)$

2. the XTS tweakable block cipher [R'04]

Let (E,D) be a secure PRP, $E: K \times \{0,1\}^n \rightarrow \{0,1\}^n$.

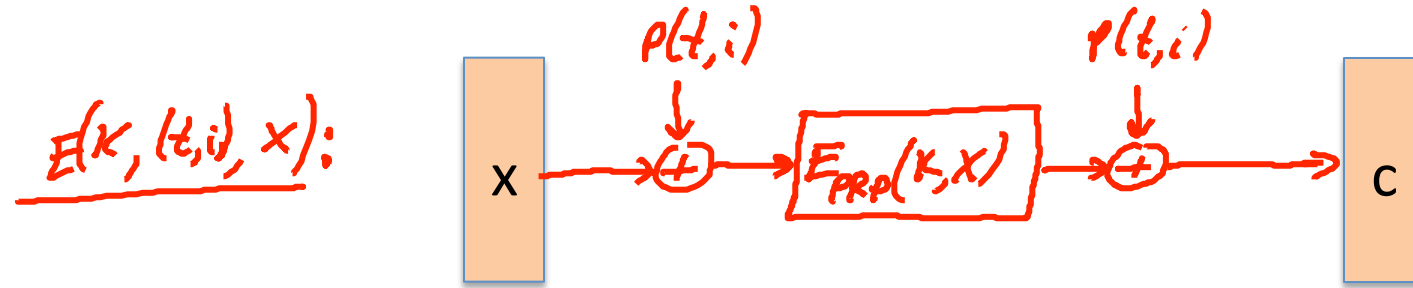
- XTS: $E_{\text{tweak}}((k_1, k_2), (t, i), x) =$



\Rightarrow to encrypt n blocks need $n+1$ evals of $E(.,.)$

Is it necessary to encrypt the tweak before using it?

That is, is the following a secure tweakable PRP?



☐ Yes, it is secure

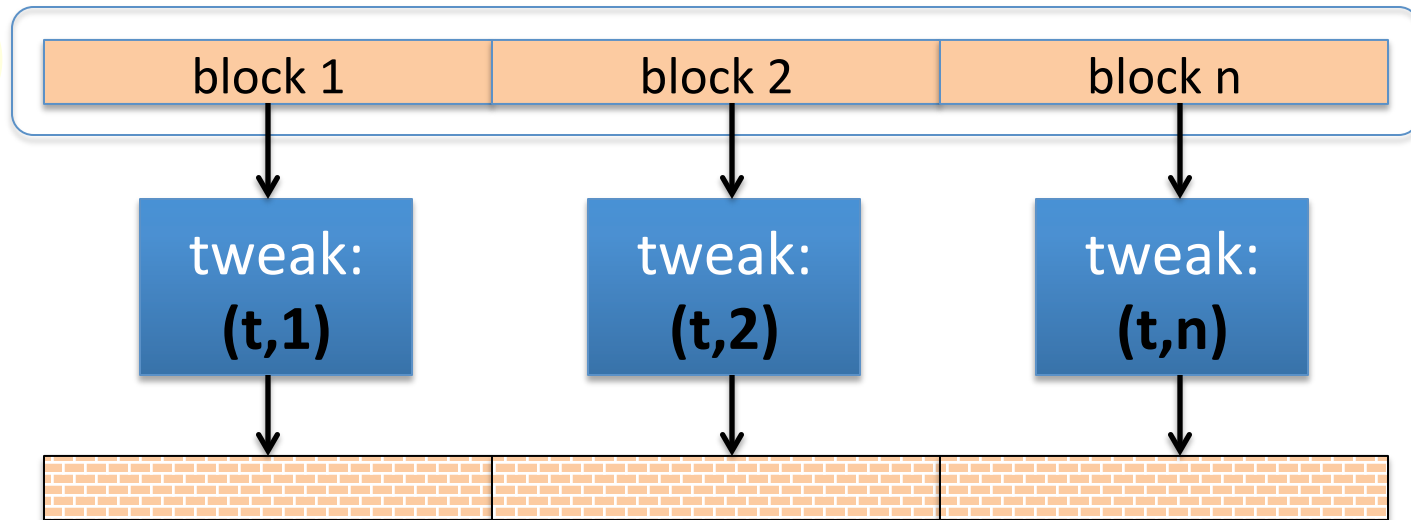
☐ No: $E(k, (t, 1), P(t, 2)) \oplus E(k, (t, 2), P(t, 1)) = P(t, 1)$

☒ No: $E(k, (t, 1), P(t, 1)) \oplus E(k, (t, 2), P(t, 2)) = P(t, 1) \oplus P(t, 2)$

☐ No: $E(k, (t, 1), P(t, 1)) \oplus E(k, (t, 2), P(t, 2)) = 0$

Disk encryption using XTS

sector # t:



- note: block-level PRP, not sector-level PRP.
- Popular in disk encryption products:

Mac OS X-Lion, TrueCrypt, BestCrypt, ...

Summary

- Use tweakable encryption when you need many independent PRPs from one key
- XTS is more efficient than the trivial construction
 - Both are narrow block: 16 bytes for AES
- EME (previous segment) is a tweakable mode for wide block
 - 2x slower than XTS

End of Segment



Odds and ends

Format preserving
encryption

Encrypting credit card numbers

Credit card format: **bbbb bnnn nnnn nnnc** (≈ 42 bits)



Goal: **end-to-end encryption**

Intermediate processors expect to see a credit card number

⇒ **encrypted credit card should look like a credit card**

Format preserving encryption (FPE)

This segment: given $0 < s \leq 2^n$, build a PRP on $\{0, \dots, s-1\}$

from a secure PRF $F: K \times \{0,1\}^n \rightarrow \{0,1\}^n$ (e.g. AES)

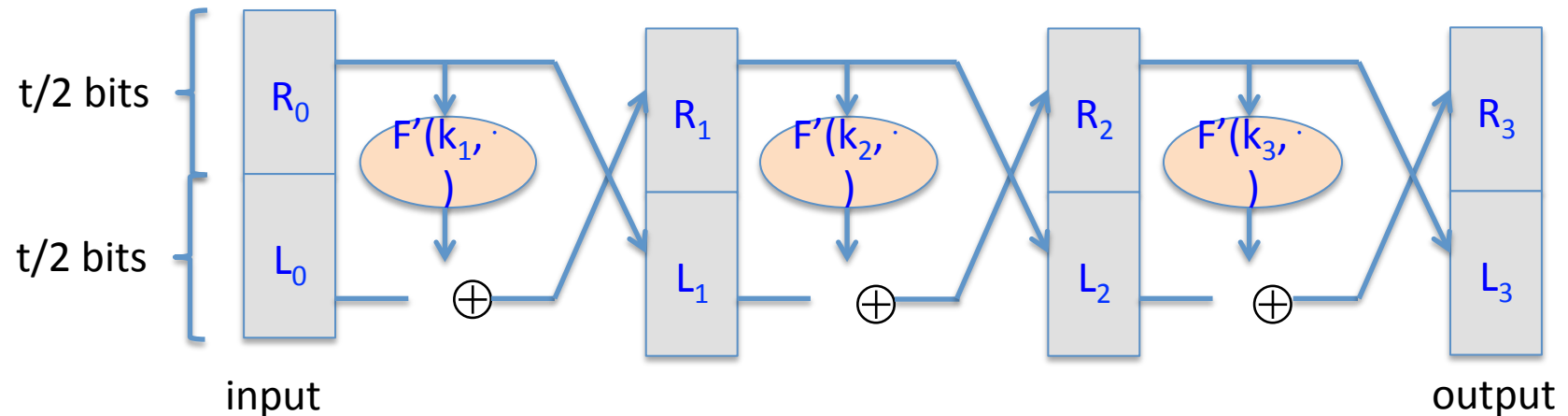
Then to encrypt a credit card number: (s = total # credit cards)

1. map given CC# to $\{0, \dots, s-1\}$
2. apply PRP to get an output in $\{0, \dots, s-1\}$
3. map output back a to CC#

Step 1: from $\{0,1\}^n$ to $\{0,1\}^t$ ($t < n$)

Want PRP on $\{0, \dots, s-1\}$. Let t be such that $2^{t-1} < s \leq 2^t$.

Method: Luby-Rackoff with $F': K \times \{0,1\}^{t/2} \rightarrow \{0,1\}^{t/2}$ (truncate F)



(better to use 7 rounds a la Patarin, Crypto'03)

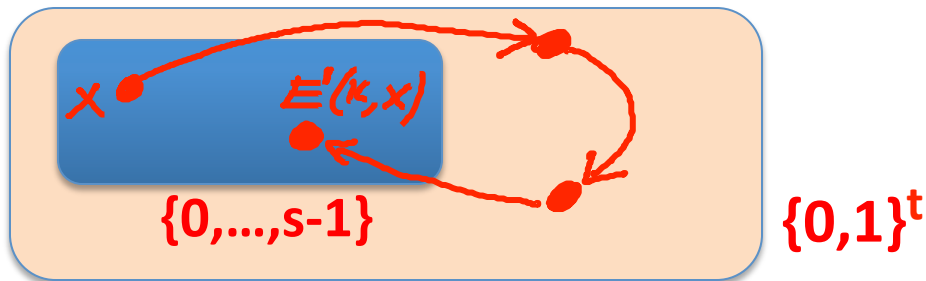
Step 2: from $\{0,1\}^t$ to $\{0,\dots,s-1\}$

Given PRP $(E,D): K \times \{0,1\}^t \rightarrow \{0,1\}^t$

we build $(E',D'): K \times \{0,\dots,s-1\} \rightarrow \{0,\dots,s-1\}$

$E'(k, x)$: on input $x \in \{0,\dots,s-1\}$ do:

$y \leftarrow x$; do $\{ y \leftarrow E(k, y) \}$ until $y \in \{0,\dots,s-1\}$; output y



Expected # iterations: 2

Security

Step 2 is tight: $\forall A \exists B: \text{PRP}_{\text{adv}}[A, E] = \text{PRP}_{\text{adv}}[B, E']$

Intuition: \forall sets $Y \subseteq X$, applying the transformation to a
random perm. $\pi: X \rightarrow X$
gives a random perm. $\pi': Y \rightarrow Y$

Step 1: same security as Luby-Rackoff construction

(actually using analysis of Patarin, Crypto'03)

note: no integrity

Further reading

- Cryptographic Extraction and Key Derivation: The HKDF Scheme.
H. Krawczyk, Crypto 2010
- Deterministic Authenticated-Encryption:
A Provable-Security Treatment of the Keywrap Problem.
P. Rogaway, T. Shrimpton, Eurocrypt 2006
- A Parallelizable Enciphering Mode. S. Halevi, P. Rogaway, CT-RSA 2004
- Efficient Instantiations of Tweakable Blockciphers and Refinements to Modes OCB and PMAC. P. Rogaway, Asiacrypt 2004
- How to Encipher Messages on a Small Domain:
Deterministic Encryption and the Thorp Shuffle.
B. Morris, P. Rogaway, T. Stegers, Crypto 2009

End of Segment