Introduction to Cryptography

F. Koeune – O. Pereira

MAT2450 – Lecture 3

Reminder: security against eavesdropper

Security experiment:

Given $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$, and adversary $\mathcal{A}$, define the following experiment $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}$:

1. $\mathcal{A}$ outputs $m_0, m_1 \in \mathcal{M}$
2. Choose $k \leftarrow \mathcal{K}$ and $b \leftarrow \{0, 1\}$, and send $\mathrm{Enc}_k(m_b)$ to $\mathcal{A}$
3. $\mathcal{A}$ outputs b'
4. Define $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi} := 1$ iff $b = b'$

Reminder: Building Encryption Schemes

Suppose G is a pseudorandom generator with expansion factor l

Let $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$ be:

- $\mathrm{Gen}(1^n)$ outputs uniformly random k from $\{0,1\}^n$
- Enc, on input $m \in \{0,1\}^{l(n)}$ and $k \in \{0,1\}^n$, provides $c := m \oplus G(k)$
- Dec, on input $c \in \{0,1\}^{l(n)}$ and $k \in \{0,1\}^n$, provides $m := c \oplus G(k)$

Multiple encryption

So far, we have seen how to encrypt one *single* message

How can we extend this to several messages?

- Repeat the same process?
  - $c_1 = G(k) \oplus m_1$
  - $c_2 = G(k) \oplus m_2$

Quizz: would that work?

1. Yes
2. No

Answer: no

- For example, anyone can compute $c_1 \oplus c_2 = m_1 \oplus m_2$

Multiple encryption

So far, we have seen how to encrypt one *single* message
How can we extend this to several messages?

- Repeat the same process?
  - $c_1 = G(k) \oplus m_1$
  - $c_2 = G(k) \oplus m_2$
  - But then, $c_1 \oplus c_2 = m_1 \oplus m_2$!
  - A very bad idea
- Use a different key for each message?
  - But how do we transmit it?
- Use a different part of the pseudorandom stream?
  - But how?

First, we need to define the security we want to achieve

Secure multiple encryption

Define the multiple-message eavesdropping experiment $\mathsf{PrivK}^{\mathsf{mult}}_{\mathcal{A},\Pi}(n)$

1. $\mathcal{A}$ outputs $M_0 = (m_0^1, \ldots, m_0^t), M_1 = (m_1^1, \ldots, m_1^t)$
2. Choose $k \leftarrow \mathrm{Gen}(1^n)$ and $b \leftarrow \{0, 1\}$, and send $(\mathrm{Enc}_k(m_b^1), \ldots, \mathrm{Enc}_k(m_b^t))$ to $\mathcal{A}$
3. $\mathcal{A}$ outputs b'
4. Define $\mathsf{PrivK}^{\mathsf{mult}}_{\mathcal{A},\Pi}(n) := 1$ iff $b = b'$

Secure multiple encryption

$\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$ has *indistinguishable multiple encryption in the presence of eavesdroppers* if
$\forall$ PPT $\mathcal{A}$, $\exists\ \epsilon$:

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

Secure multiple encryption

Does secure (single) encryption imply secure multiple encryption?

- Of course not! (see previous attempt)

But can we prove it?

- That is, taking the naive "repetition" idea of slide 5, can we build an adversary that wins $\mathsf{PrivK}^{\mathsf{mult}}_{\mathcal{A},\Pi}$ with non-negligible probability?

- Easy example: $\mathcal{A}$ outputs

$$M_0 = (0 \ldots 0, 0 \ldots 0), M_1 = (0 \ldots 0, 1 \ldots 1)$$

Probabilistic encryption

Observation: we cannot achieve acceptable security with a *deterministic* scheme[1]

We need *probabilistic encryption*

- The same message, encrypted with the same key, yields different results
- Of course, decryption remains deterministic

[1] At least, not without maintaining a state between encryptions.

Remark

Encrypting the same message twice is not the only problem
As we have seen, if we encrypt two *different* messages as

- $c_1 = G(k) \oplus m_1$
- $c_2 = G(k) \oplus m_2$

then the adversary learns $c_1 \oplus c_2 = m_1 \oplus m_2$

This is a lot of information!

- If one of the messages is (partially) known
- If both messages are in English
- ...

A frequent and devastating mistake

Security against Chosen-Plaintext Attacks (CPA)

So far, we have only considered *passive* adversaries

- eavesdrops on ciphertext
- must recover (some info on) plaintext

But a real-world adversary could have access to additional information

- previous encryptions (with same key) of messages he knows
- previous encryptions (with same key) of messages he has chosen
- ...

Can we capture these notions?

The new adversary

Let us define a more powerful adversary

- Granted access to an *encryption oracle* $\mathrm{Enc}_k(\cdot)$ that will encrypt messages of his choice
- Allowed to call oracle adaptively, before and after submitting two challenge messages m_0, m_1 of his choice
- As before, must tell whether he receives $\mathrm{Enc}_k(m_0)$ or $\mathrm{Enc}_k(m_1)$

More formally...

Given $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$, and adversary $\mathcal{A}$, define the following experiment $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$:

1. Choose $k \leftarrow \mathrm{Gen}(1^n)$
2. $\mathcal{A}$ is given oracle access to $\mathrm{Enc}_k(\cdot)$
3. $\mathcal{A}$ outputs $m_0, m_1 \in \mathcal{M}$
4. Choose $b \leftarrow \{0, 1\}$ and send $\mathrm{Enc}_k(m_b)$ to $\mathcal{A}$
5. $\mathcal{A}$ is again given oracle access to $\mathrm{Enc}_k(\cdot)$
6. $\mathcal{A}$ outputs b'
7. Define $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) := 1$ iff $b = b'$

Security against CPA

$\Pi := \langle \text{Gen}, \text{Enc}, \text{Dec} \rangle$ has *indistinguishable encryption under a chosen-plaintext attack* if $\forall$ PPT $\mathcal{A}$, $\exists\ \epsilon$:

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

Remark: what is the relationship between $\mathsf{PrivK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$ and $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$?

- CPA-security $\Rightarrow$ security against an eavesdropper

Quizz: why not this?

Given $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$, and adversary $\mathcal{A}$, define the following experiment $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$:

1. Choose $k \leftarrow \mathrm{Gen}(1^n)$
2. $\mathcal{A}$ is given oracle access to $\mathrm{Enc}_k(\cdot)$
3. $\mathcal{A}$ outputs $m_0, m_1 \in \mathcal{M}$
4. Choose $b \leftarrow \{0, 1\}$ and send $\mathrm{Enc}_k(m_b)$ to $\mathcal{A}$
5. $\mathcal{A}$ is again given oracle access to $\mathrm{Enc}_k(\cdot)$, but cannot ask for $\mathrm{Enc}_k(m_0)$ or $\mathrm{Enc}_k(m_1)$
6. $\mathcal{A}$ outputs b'
7. Define $\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) := 1$ iff $b = b'$

Quizz

Why don't we prevent the adversary from choosing m_0 or m_1 after the challenge phase ?

Answer:

- Because we really want this case to be taken into account, and ensure that the adversary will lose, even then.
- We want an encryption scheme that prevents from recognizing whether two ciphertexts correspond to the same plaintext (i.e. probabilistic encryption).

Extending the definition

As before, we can extend this notion to multiple encryption

Definition extension is straightforward (try it!)

Good news:

- CPA security for single encryption $\Rightarrow$ CPA security for multiple encryption

Relation between definitions

We have:

$$\text{CPA} \Rightarrow \text{Multiple message eavesdropper} \Rightarrow \text{Eavesdropper}$$
$$\text{CPA} \not\Leftarrow \text{Multiple message eavesdropper} \not\Leftarrow \text{Eavesdropper}$$

How to perform CPA-secure encryption?

Idea for a probabilistic encryption scheme:
Change Enc as follows:

- Pick $r \leftarrow \{0,1\}^n$ and encrypt m as $\langle r, G(k\|r) \oplus m \rangle$

Does it work?

- G only guaranteed to output pseudorandom values with *secret* seed.
- Let G be a PRG. Define the PRG $G'(s_1\|s_2) = s_1\|G(s_2)$:
  - It is still length increasing
  - If the output of G is pseudorandom, then so is the one of G'.

But using G' would leak k immediately!

How to perform CPA-secure encryption?

We need something stronger than just a PRG:

We need a F such that:

- When r is public (but not k), $F(k, r)$ is pseudorandom
- For two randomly-chosen public r_1, r_2,
  $F(k, r_1)$ and $F(k, r_2)$ are pseudorandom

Pseudorandom functions

First, what is a random function?

- A function can be described as a big (input,output) table
- A random function is one such table, with all outputs chosen at random

A pseudorandom function is one that cannot be efficiently distinguished from a truly random one

- Does not make much sense to say that a *fixed* function is pseudorandom
- Instead, we will consider *distributions* on functions, using *keyed functions*

Pseudo-random functions

A keyed function F is a function

$$F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* : (k,x) \to F(k,x)$$

F is said to be efficient if there is a deterministic polynomial-time algorithm to compute $F(k,x)$ given k,x

A keyed function introduces a distribution on functions

- Choose a random k
- Define $F_k(x) := F(k,x)$

We say that F is pseudorandom if, for random k, F_k cannot be distinguished from a random function
Note: we will focus on *length-preserving* functions

How dense is the distribution of F_k?

A random function $\{0,1\}^n \to \{0,1\}^n$ can be described as a sorted table of outputs

- 2^n entries in the table
- Each entry is n bit long
- Full table has size $n.2^n$ bits

Conversely, each such table describes a valid function
$\Longrightarrow$There are thus $2^{n.2^n}$ possible functions

On the other hand, for a given F, there are only 2^n possible functions $F_k(\cdots)$

$\Longrightarrow F$ generates only a very small part of the full space

Indistiguishability

How can we formalize the notion "indistinguishable from a random function"?

- Attempt 1: a distinguisher D receiving a challenge function g cannot, in polynomial time, tell whether g is a true random function f or a PRF F_k for some k
- Problem: describing g requires $n.2^n$ bits: this is not polynomial
- So D cannot "read" g in polynomial time

Instead, we will give D *oracle access* to g

- D can query g with values x of his choice and get corresponding results $g(x)$

Indistiguishability

$F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^*$ is a pseudorandom function if $\forall$ PPT D, $\exists$ negl. ϵ:

$$\left|\Pr[D^{F_k(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1]\right| \leq \epsilon(n)$$

Remarks

- k is chosen uniformly at random in $\{0,1\}^n$
- D is not given the key k
- As D is PPT, he can only do a polynomially bounded number of oracle queries

Do pseudorandom functions exist?

In fact, we do not know

- It has been shown that PRF functions exist iff pseudorandom generators exist
- In practice, we have good candidates (discussed later)

Building a CPA-secure scheme from a PRF

Define $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$ as:

- Gen: choose random $k \leftarrow \{0,1\}^n$
- Enc: on input $m, k \in \{0,1\}^n$,
  - choose random $r \leftarrow \{0,1\}^n$
  - $c := \langle r, F_k(r) \oplus m \rangle$
- Dec: on input $k \in \{0,1\}^n$ and $c = \langle r, s \rangle$, output $m := s \oplus F_k(r)$

Security of this construction

Theorem: if F is a pseudorandom function, this construction has indistinguishable encryption under a chosen-plaintext attack

Proof: in two steps:

- Prove that the scheme is secure if F_k is replaced by a truly random function
- Prove that if the scheme (with F_k) were insecure, we could distinguish F_k from a truly random function

Step 1: idealized scheme

Initial scheme:

Define $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$ as:

- Gen: choose random $k \leftarrow \{0,1\}^n$
- Enc: on input $m, k \in \{0,1\}^n$,
  - choose random $r \leftarrow \{0,1\}^n$
  - $c := \langle r, F_k(r) \oplus m \rangle$
- Dec: on input $k \in \{0,1\}^n$ and $c = \langle r, s \rangle$, output $m := s \oplus F_k(r)$

Step 1: idealized scheme

Idealized version

Define $\tilde{\Pi} := \langle \widetilde{\mathrm{Gen}}, \widetilde{\mathrm{Enc}}, \widetilde{\mathrm{Dec}} \rangle$ as:

- $\widetilde{\mathrm{Gen}}$: choose random f
- $\widetilde{\mathrm{Enc}}$: on input $m, k \in \{0,1\}^n$,
  - choose random $r \leftarrow \{0,1\}^n$
  - $c := \langle r, f(r) \oplus m \rangle$
- $\widetilde{\mathrm{Dec}}$: on input $k \in \{0,1\}^n$ and $c = \langle r, s \rangle$, output $m := s \oplus f(r)$

This is *almost* a one-time pad

Step 1: idealized scheme

Consider a CPA-adversary $\mathcal{A}$

- Uses oracle $\widetilde{\mathrm{Enc}}_k(\cdot)$ to encrypt messages of his choice
- Outputs m_0, m_1
- Receives $\widetilde{\mathrm{Enc}}_k(m_b) = \langle r, f(r) \oplus m_b \rangle$ with random b
- Uses oracle $\widetilde{\mathrm{Enc}}_k(\cdot)$ to encrypt messages of his choice
- Must tell whether $b = 0$ or 1

(Note: $\mathcal{A}$ makes at most $q(n)$ oracle queries in total)

What are his chances of success?

Chances of success

Two cases:

- Case 1: r is never used by oracle except for m_b
  - Then $\mathcal{A}$ learns nothing useful from oracle queries
    - (f random $\Rightarrow f(r) \perp f(r')$ for $r \neq r'$)

Quizz: What are $\mathcal{A}$'s chances of success then?

1. $\frac{1}{2}$
2. $\frac{1}{2} + \epsilon(n)$
3. $\epsilon(n)$

Answer: 1

- This corresponds to a OTP

Chances of success

Two cases:

- Case 1: r is never used by oracle except for m_b
  - Then $\mathcal{A}$ learns nothing useful from oracle queries
    - (f random $\Rightarrow f(r) \perp f(r')$ for $r \neq r'$)
  - Same as one-time pad
  - $\mathcal{A}$ succeeds with probability $\frac{1}{2}$

- Case 2: r has been used at least once by oracle
  - $\mathcal{A}$ has $\langle r, f(r) \oplus m_b \rangle$ and $\langle r, f(r) \oplus m' \rangle$, for some m' he knows
  - $\mathcal{A}$ can easily deduce $f(r)$ and hence the value of b
  - $\mathcal{A}$ always wins in this case
  - But the probability for this case is only $\frac{q(n)}{2^n}$

Chances of success

So,

$$
\begin{aligned}
\Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\tilde{\Pi}}(n) = 1] &= \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\tilde{\Pi}}(n) = 1 \wedge \mathsf{Repeat}] \\
&\quad + \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\tilde{\Pi}}(n) = 1 \wedge \overline{\mathsf{Repeat}}] \\
&\leq \Pr[\mathsf{Repeat}] \\
&\quad + \Pr[\mathsf{PrivK}^{\mathsf{cpa}}_{\mathcal{A},\tilde{\Pi}}(n) = 1 | \overline{\mathsf{Repeat}}] \\
&\leq \frac{q(n)}{2^n} + \frac{1}{2}
\end{aligned}
$$

We have thus showed that, if a truly random f is used, the scheme is secure

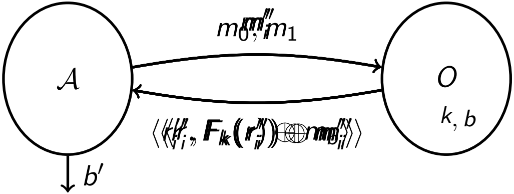
Step 2: real scheme is as good as idealized one

Suppose that there exists $\mathcal{A}$ that can break the (real) scheme with non-negligible probability $\frac{1}{2} + \eta(n)$

We will show that $\mathcal{A}$ can distinguish F_k from a truly random function with non-negligible probability

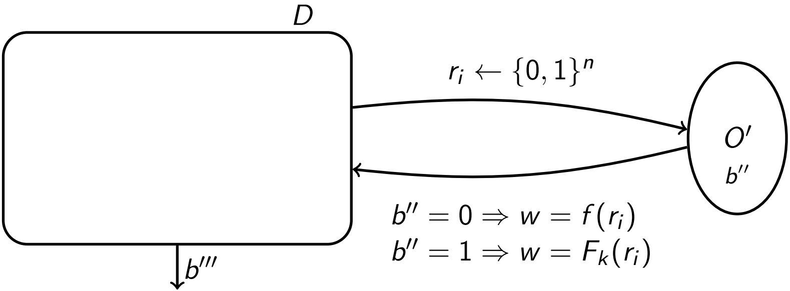
What does the adversary look like?

Query phase 1
Challenge phase
Query phase 2
Answer phase



By assumption, $\Pr[b = b'] = \frac{1}{2} + \eta(n)$, with η non-negligible.

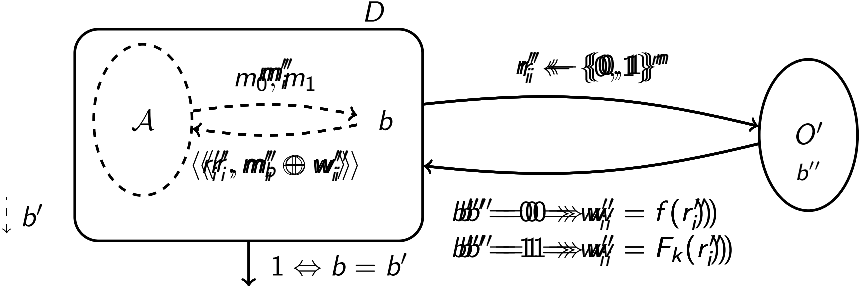
What distinguisher must we build?



Query phase
Answer phase

Reduction

Query phase 2
(identical to query phase 1)



Step 2: real scheme is as good as idealized one

On the one hand,

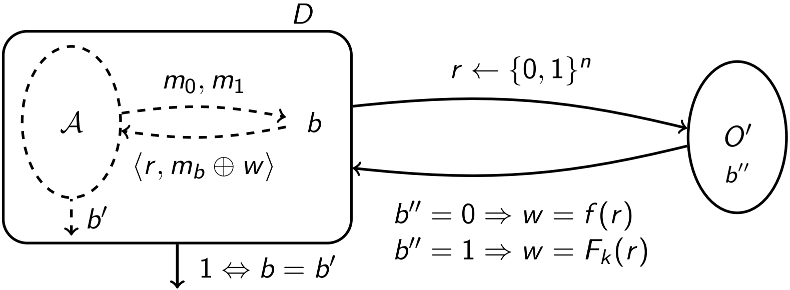
- If O' is the function F_k, then $\mathcal{A}$ is de facto interfaced with $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$
- By assumption, $\mathcal{A}$ will then win with non-negligible probability

On the other hand,

- If O' is a random f, then $\mathcal{A}$ is de facto interfaced with $\widetilde{\Pi} := \langle \widetilde{\mathrm{Gen}}, \widetilde{\mathrm{Enc}}, \widetilde{\mathrm{Dec}} \rangle$
- And we have showed that, in this case, $\mathcal{A}$ *cannot* win with non-negligible probability

$\Longrightarrow$ Let us run $\mathcal{A}$ and see whether it wins

Reduction



So:

- If $b'' = 0$, $\Pr[D \to 1] \leq \frac{1}{2} + \frac{q(n)}{2^n}$: idealized scheme
- If $b'' = 1$, $\Pr[D \to 1] = \frac{1}{2} + \eta(n)$: normal security game
- D distinguishes with $\Pr \geq \eta(n) - \frac{q(n)}{2^n}$

Distinguisher

So,

$$\left|\Pr[D^{F_k(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1]\right| \geq \eta(n) - \frac{q(n)}{2^n}$$

And, since F is a PRF, $\eta(n) - \frac{q(n)}{2^n}$ must be negligible,
so, $\eta(n)$ must be negligible

Summary

We showed that

- If we use a truly random function in our construction, then the scheme is secure
- If someone can break the scheme with a practical function F_k, then he can distinguish F_k from a truly random function

$\Longrightarrow$ If F_k is a PRF, the scheme is secure

Multiple-Message Security

CPA security implies multiple-message security

- But focuses on the confidentiality of *one* message

(Reminder) Our central idea to randomize: encrypt each message as $c := \langle r, F_k(r) \oplus m \rangle$ with a random r

When we encrypt l messages, what is the probability that we draw the same r more than once?

- "How many messages can we encrypt before being at risk?"

The birthday paradox

"How many people do we need in a room to have a probability of at least one half that two of them have the same birth date?"

- Answer: 23
- For many people, this is surprizingly low
- This is known as the *birthday paradox*
- It can be roughly generalized as follows: If we take q random values from a space of size N, the probability of collision is about $\frac{q^2}{2N}$
- So, more simply: If we take random values from a space of size N, we might expect a collision after about $\sqrt{N}$ values

Birthday paradox and security

So, the risk to "draw" the same r twice is driven by the birthday paradox

Negligible if the number of messages is negligible compared to $2^{n/2}$ (where $n =$ block length)

- Number of blocks is polynomial, $2^{n/2}$ is exponential

$\Rightarrow$ OK

Remarks:

- means that not only the key size, but also the *block size* is a security factor – DES had blocks of size 64 bits!
- AES, the most widely used PRF today, has blocks of size 128 bits
  Independently of the key size!

Efficiency

OK, the construction

- choose random $r \leftarrow \{0,1\}^n$
- $c := \langle r, F_k(r) \oplus m \rangle$

is secure, but it is not very efficient (doubles length!)

Can we do better?

Modes of Operation

PRF (and PRPs, see next) are typically used in a *mode of operation*:

- ways to use a fixed length PRF/PRP in order to offer security for messages of arbitrary length.

CTR Mode

Let F be a fixed-length PRF: $\{0,1\}^n \times \{0,1\}^n \rightarrow \{0,1\}^n$

Define $\Pi := \langle \mathrm{Gen}, \mathrm{Enc}, \mathrm{Dec} \rangle$ as:

- Gen: choose random $k \leftarrow \{0,1\}^n$
- Enc: on input $k \in \{0,1\}^n$, $m = \langle m_1, \ldots, m_l \rangle \in (\{0,1\}^n)^l$
  - choose random $r \leftarrow \{0,1\}^n$
  - $c := \langle r, c_1, \ldots, c_l \rangle$ with $c_i = F_k(r+i) \oplus m_i$ (compute sums mod 2^n)
- Dec: proceed in the natural way

Observe:

- $F_k(r+i)$ can be precomputed: encryption is then very fast
- Encryption can be made parallel: good for multi-core
- Easy to decrypt only block c_i: good for HDD encryption
- if m_l is not full length, just truncate the output of $F_k(r+l)$

CTR Mode Security

If F is a PRF then CTR mode offers CPA security.

- $F_k(r+1) \| F_k(r+2) \| \cdots \| F_k(r+l)$ is a pseudorandom stream, even if r is public
- Security proof follows previous single-block proof
  Main difference: bound on collision of $r_1 + i_1$ and $r_2 + i_2$

  If $q(n)$ queries of max length $q(n)$ blocks
  Proba that $\mathrm{Enc}_k(m_b)$ has $r + i$ overlapping another block is $\leq 2q(n)^2/2^n$

CTR is used to encrypt in TLS 1.2 and TLS 1.3 (with authentication)

Pseudorandom permutations and block ciphers

We have introduced the notion of pseudorandom function

Similarly, we can also define pseudorandom *permutations* (PRP), i.e. one-to-one functions

Most of the time PRPs are used for PRFs in practice!

Pseudorandom permutations and block ciphers

A keyed permutation F is a function

$$F : \{0,1\}^* \times \{0,1\}^* \to \{0,1\}^* : (k,x) \to F(k,x)$$

such that, $\forall k$, $F_k(\cdot)$ is one-to-one

F is said to be efficient if there is

- a deterministic polynomial-time algorithm to compute $F_k(x)$ given k, x
- *and* a deterministic polynomial-time algorithm to compute $F_k^{-1}(x)$ given k, x

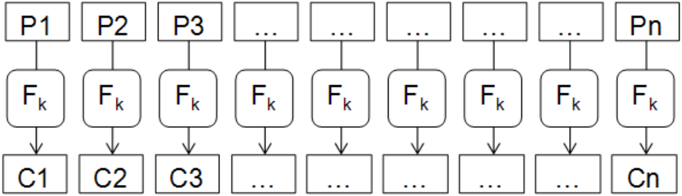
We say that F is pseudorandom if, for random k, F_k cannot be distinguished from a random permutation

How can we encrypt with a PRP?

The bad idea (why?): $c := F_k(m)$

This cannot be CPA-secure

Electronic codebook (ECB)

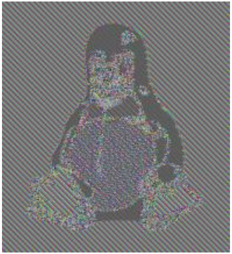
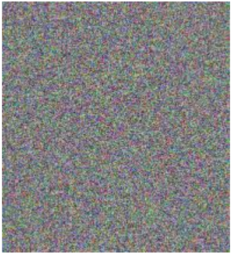


Basic mode: no mode of operation

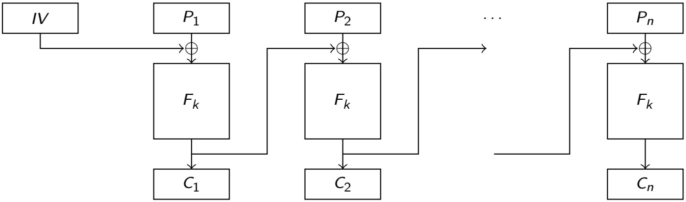
Not CPA-secure

Not even secure against an eavesdropper

ECB is not secure

This	yields this	and not this
		

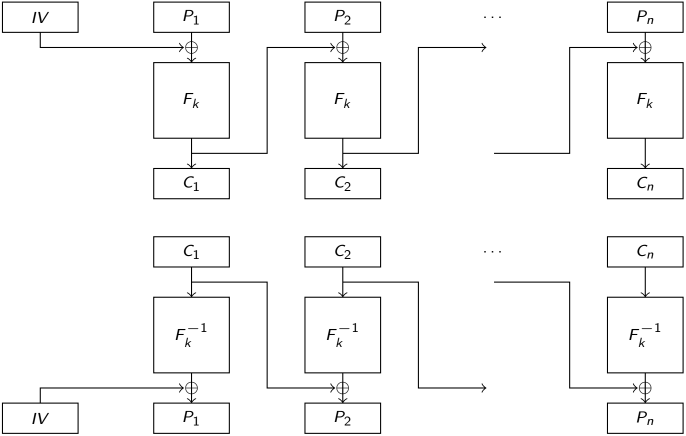
Cipher block chaining (CBC)



CPA-secure, provided F_k is a PRP

Note: in modes of operation, the random value r is traditionally called the Initialization Vector (IV)

CBC : how do we decrypt?



CBC-mode

Observations:

- Less efficient: no precomputation, not parallelizable
- IV (near-)collisions less disastrous:
  different m_i's cause divergence
- Can also serve for authentication (with fixed IV)
  (see next week)

Uses:

- TLS 1.0, 1.1, 1.2

Modes of operation: summary

Constructions allowing to build secure encryption schemes based on PRPs (building blocks)

Can be proved secure (except ECB!) provided building block is secure

We only saw a few of them (also CFB, OFB. . .)