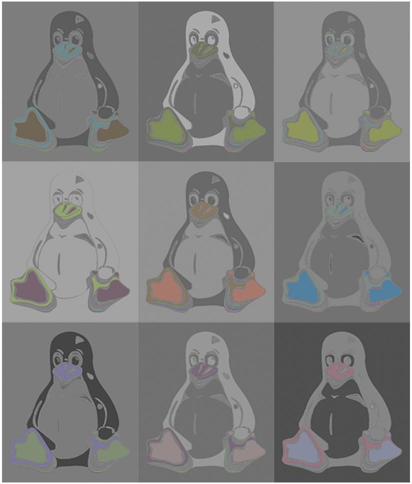


THE ECB PENGUIN

CS 553

CRYPTOGRAPHY

Lecture 15

Modes of Operation

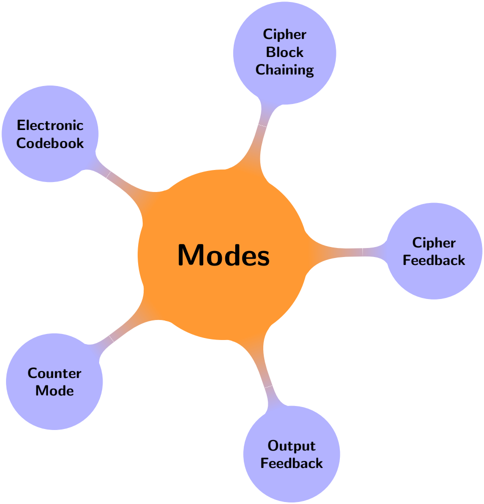
Instructor
Dr. Dhiman Saha

https://blog.filippo.io/the-ecb-penguin/

Modes of Operation

The Domain Extension Algorithm

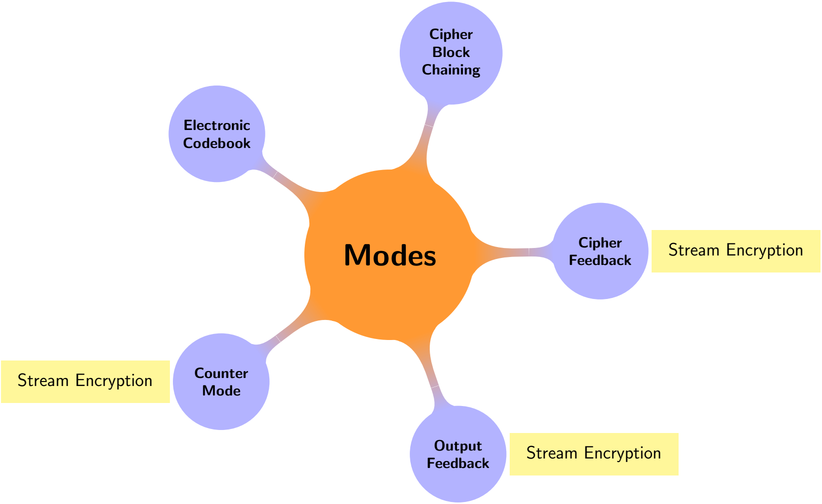
Handling arbitrary amount of data using a fixed length function



Modes of Operation

The Domain Extension Algorithm

Handling arbitrary amount of data using a fixed length function



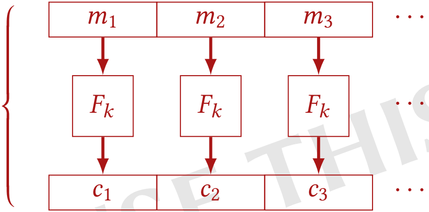
ECB — The Electronic Codebook

Enc$(k, m_1 \cdots m_\ell)$:

for $i = 1$ to ℓ:

$\quad c_i := F(k, m_i)$

return $c_1 \cdots c_\ell$

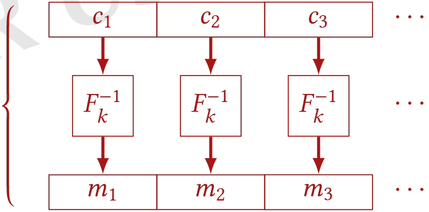


Dec$(k, c_1 \cdots c_\ell)$:

for $i = 1$ to ℓ:

$\quad m_i := F^{-1}(k, c_i)$

return $m_1 \cdots m_\ell$



NEVER USE THIS!

ECB In general not recommended!

- Same plaintext always encrypts to same ciphertext

Recall ECB not IND-CPA Secure

Indistinguishability under Chosen Plaintext Attack

Possible use-cases for ECB

- Encrypting only single-block messages
- Using random characters with input blocks
- Reduces capacity of each block
- Must be undone during decryption

Advantages

- Parallelizable Encryption Random Access Decryption

Transmission errors $\implies$ Incorrect Ciphertext

Bit Flip ($0 \to 1 \mid 1 \to 0$) Bit Drop ($0101 \to 001$)

Bit Flip $\implies$ Bad plaintext block after decryption

- Limited to **one** block only
- Other blocks unaffected

Bit Drop $\implies$ alignment lost

- All plaintext will be bad after decryption
- Fixed only if alignment recovered

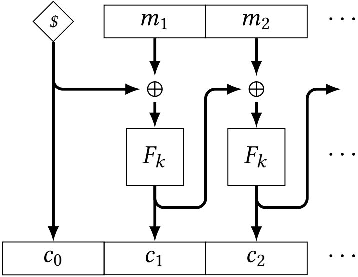
$\underline{\text{Enc}(k, m_1 \cdots m_\ell):}$

$c_0 \leftarrow \{0,1\}^{blen}$;

for $i = 1$ to ℓ:

$\quad c_i := F(k, m_i \oplus c_{i-1})$

return $c_0 c_1 \cdots c_\ell$

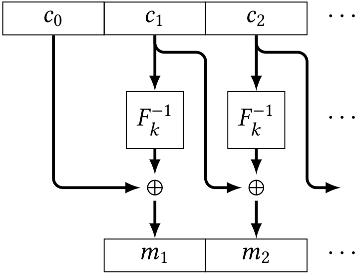


$\underline{\text{Dec}(k, c_0 \cdots c_\ell):}$

for $i = 1$ to ℓ:

$\quad m_i := F^{-1}(k, c_i) \oplus c_{i-1}$

return $m_1 \cdots m_\ell$



CBC — Random IV

- Encryption cannot be parallelized
- No random access
- Each ciphertext block depends on all the previous blocks

IV — The Initialization Vector

Random IV $\implies$ **Randomized** encryption $\implies$ IND-CPA

- Identical plaintexts will encrypt to **distinct** ciphertexts
- When calling the cipher with **distinct** IVs

Note: IV must be communicated for decryption

- Encrypting l blocks under CBC mode results in $l + 1$ blocks

What if constant IV is used?

Common prefix leakage

CBC — Leakage with Random IV?

Assume

$c_i = c_j$ for some $1 \leq i, j \leq n$ with $i \neq j$

$$
\begin{aligned}
c_i = c_j &\implies F(k, m_i \oplus c_{i-1}) = F(k, m_j \oplus c_{j-1}) \\
&\implies (m_i \oplus c_{i-1}) = (m_j \oplus c_{j-1}) \\
&\implies (m_i \oplus m_j) = (c_{i-1} \oplus c_{j-i}) \quad \leftarrow \text{Info. leakage}
\end{aligned}
$$

For $b-$bit block cipher — The Birthday Paradox[1]

$$\Pr\left[c_i = c_j\right] \approx 2^{\frac{b}{2}}$$

- Justifies need for **larger** block size

[1] Will be detailed in hash function lecture

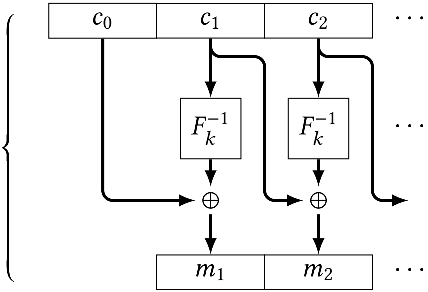
CBC — Can decryption be parallelized?

$\underline{\mathrm{Dec}(k, c_0 \cdots c_\ell):}$

for $i = 1$ to ℓ:

  $m_i := F^{-1}(k, c_i) \oplus c_{i-1}$

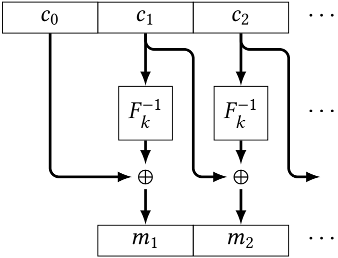
return $m_1 \cdots m_\ell$



- c_i computation depends on c_{i-1} (Encryption)
- m_i computation **does not** depends on m_{i-1} (Decryption)
- It depends on c_i, c_{i-1}
- Parallel computation **possible** if all previous c_i available (generally true)

A single-bit transmission error in ciphertext block $c_i \implies$

- Whole plaintext block m_i corrupted
- The same bit in plaintext block m_{i+1} being corrupted



Self-Recovering/Self-Synchronizing Property

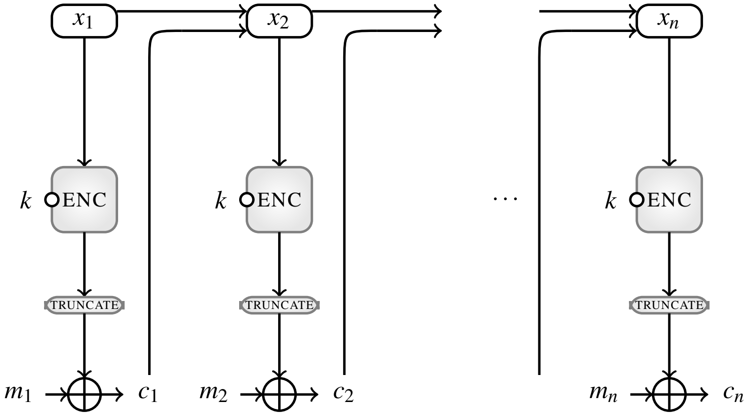
Blocks following m_{i+1} will not be affected

If a bit is added or lost from the cipher-text stream, then all subsequent blocks are garbled.

Stream Encryption

CFB · OFB · CTR

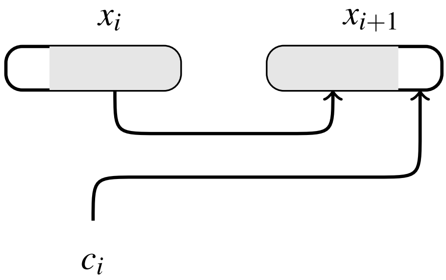
A stream cipher out of a block cipher



- t bits encrypted with each call to the block cipher[2]

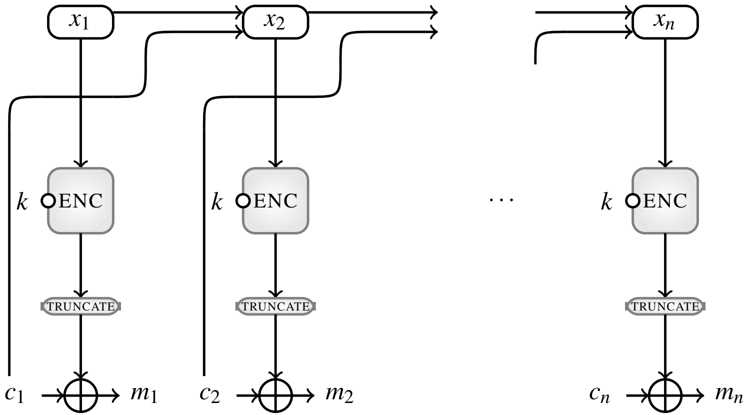
$$c_i = m_i \oplus MSB_t(ENC_k(x_i)), \quad \begin{cases} x_1 = \text{chosen } IV \\ x_{i+1} = LSB_{b-t}(x_i) || c_i \end{cases}$$

[2] $(1 \leq t \leq b)$ and $(1 \leq i \leq n)$



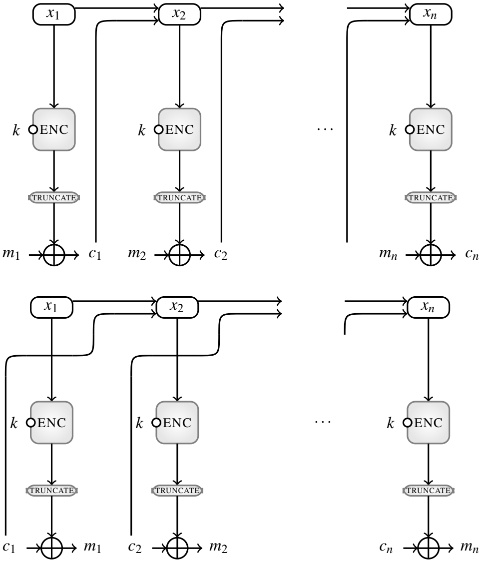
Update of state register from x_i to x_{i+1}

- First $b - t$ bits of x_i are shifted to the left
- Then c_i is used to replace the missing bits on the right



- Decryption has a similar form to encryption

$$m_i = c_i \oplus MSB_t(ENC_k(x_i)), \quad \begin{cases} x_1 = IV \text{ used in encryption} \\ x_{i+1} = LSB_{b-t}(x_i) \| c_i \end{cases}$$



- Only forward encryption required
- Not parallelizable
- One call per $t-$bits of ciphertext

CFB Mode

Block Cipher $\Longrightarrow$ Self-synchronizing stream cipher

- An error in some CFB-encrypted ciphertext block c_i will be inherited by the corresponding plaintext block m_i that is recovered
- Since x_{i+1} will contain the incorrect c_i, the recovery of subsequent message blocks will be garbled until the source register x_j for some $j > i$ is free from the influence of c_i
- This will happen when c_i has been shifted out of the register
- #plaintext-blocks corrupted by single ciphertext error

$$\leq \left\lceil \frac{b}{t} \right\rceil + 1$$

- So, provided sufficiently many ciphertext bits are received without error, correct decryption can be recovered.

CFB Mode

Block Cipher $\Longrightarrow$ Self-synchronizing stream cipher

- An error in some CFB-encrypted ciphertext block c_i will be inherited by the corresponding plaintext block m_i that is recovered
- Since x_{i+1} will contain the incorrect c_i, the recovery of subsequent message blocks will be garbled until the source register x_j for some $j > i$ is free from the influence of c_i
- This will happen when c_i has been shifted out of the register
- #plaintext-blocks corrupted by single ciphertext error

$$\leq \left\lceil \frac{b}{t} \right\rceil + 1$$

- So, provided sufficiently many ciphertext bits are received without error, correct decryption can be recovered.

CFB Mode

Block Cipher $\implies$ Self-synchronizing stream cipher

- An error in some CFB-encrypted ciphertext block c_i will be inherited by the corresponding plaintext block m_i that is recovered
- Since x_{i+1} will contain the incorrect c_i, the recovery of subsequent message blocks will be garbled until the source register x_j for some $j > i$ is free from the influence of c_i
- This will happen when c_i has been shifted out of the register
- #plaintext-blocks corrupted by single ciphertext error

$$\leq \left\lceil \frac{b}{t} \right\rceil + 1$$

- So, provided sufficiently many ciphertext bits are received without error, correct decryption can be recovered.

CFB Mode

Block Cipher $\Longrightarrow$ Self-synchronizing stream cipher

- An error in some CFB-encrypted ciphertext block c_i will be inherited by the corresponding plaintext block m_i that is recovered
- Since x_{i+1} will contain the incorrect c_i, the recovery of subsequent message blocks will be garbled until the source register x_j for some $j > i$ is free from the influence of c_i
- This will happen when c_i has been shifted out of the register
- #plaintext-blocks corrupted by single ciphertext error

$$\leq \left\lceil \frac{b}{t} \right\rceil + 1$$

- So, provided sufficiently many ciphertext bits are received without error, correct decryption can be recovered.

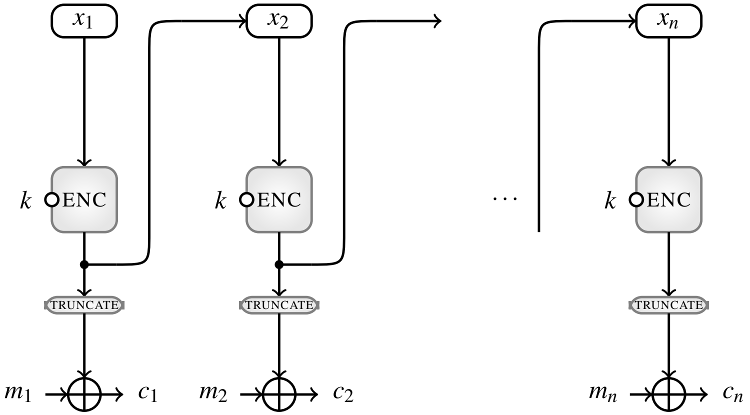
CFB Mode

Block Cipher $\Longrightarrow$ Self-synchronizing stream cipher

- An error in some CFB-encrypted ciphertext block c_i will be inherited by the corresponding plaintext block m_i that is recovered
- Since x_{i+1} will contain the incorrect c_i, the recovery of subsequent message blocks will be garbled until the source register x_j for some $j > i$ is free from the influence of c_i
- This will happen when c_i has been shifted out of the register
- #plaintext-blocks corrupted by single ciphertext error

$$\leq \left\lceil \frac{b}{t} \right\rceil + 1$$

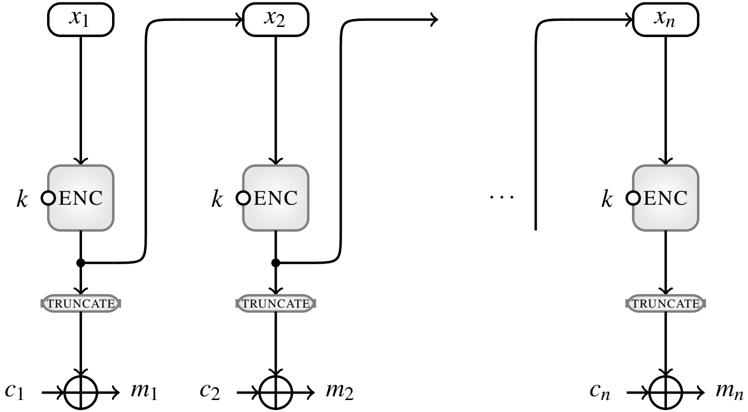
- So, provided sufficiently many ciphertext bits are received without error, correct decryption can be recovered.



- t bits encrypted with each call to the block cipher[3]

$$c_i = m_i \oplus MSB_t(ENC_k(x_i)), \quad \begin{cases} x_1 = \text{Chosen } IV \\ x_{i+1} = ENC_k(x_i) \end{cases}$$

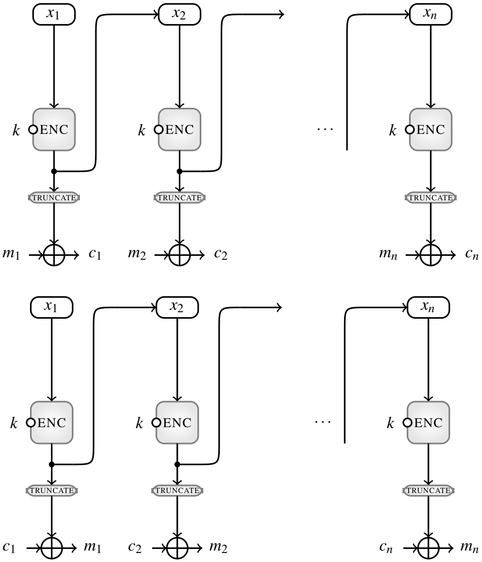
[3] $(1 \leq t \leq b)$ and $(1 \leq i \leq n)$



- Decryption has a similar form to encryption

$$m_i = c_i \oplus MSB_t(ENC_k(x_i)), \quad \begin{cases} x_1 = \text{Chosen } IV \\ x_{i+1} = ENC_k(x_i) \end{cases}$$

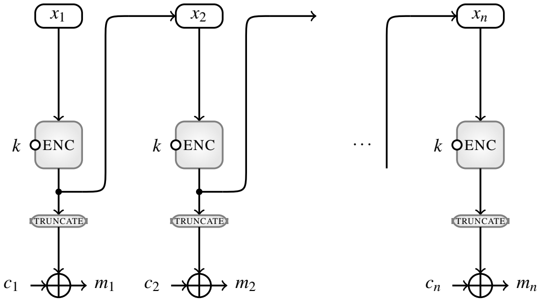
The Whole Picture



- Only forward encryption required
- Not parallelizable
- Stream-cipher mode
- But here encryption does not depend on previous ciphertexts.

Note: (key, IV) pair should not be reused

Fault Tolerance



Bit Errors

- Affect corresponding plaintext
- But no error propagation

Bit Loss

- Leads to alignment loss
- Require external resynchronization

CFB Vs OFB

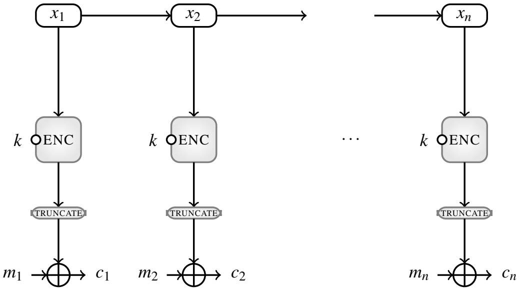
CFB

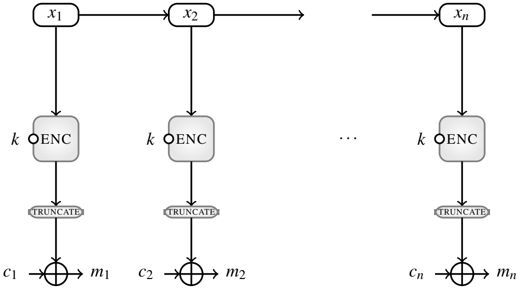
- Stream Cipher
- **Self-synchronizing**
- Key-stream is dependent on message or ciphertext

OFB

- Stream Cipher
- **Synchronous**
- Key-stream is independent of message or ciphertext

CTR (Parallelizable) Counter Mode





- Almost similar to OFB w.r.t fault tolerance
- Also a Synchronous Stream Cipher Mode

Main Difference with OFB

- CTR supports random access for decryption
- CTR supports parallel encryption

IV Selection   Needed by all CBC, CFB, OFB, and CTR

OFB and CTR

- Same (Key,IV) $\implies$ Same key-stream!!!
- IV can be known to some adversary, but used only **once**
- Also referred to as nonce
- Can be a counter

CBC and CFB

- Require the IV to be unpredictable. Why? (HomeWork)
- Notion of Pre-IV
- IV $= ENC_k$(Pre-IV)
- IV $= ENC_{k'}$(Pre-IV)

IV Selection — Needed by all CBC, CFB, OFB, and CTR

OFB and CTR

- Same (Key,IV) $\implies$ Same key-stream!!!
- IV can be known to some adversary, but used only **once**
- Also referred to as nonce
- Can be a counter

CBC and CFB

- Require the IV to be unpredictable. Why? (HomeWork)
- Notion of Pre-IV
- IV $= ENC_k(\text{Pre-IV})$
- IV $= ENC_{k'}(\text{Pre-IV})$

Padding

- Not a concern for OFB and CTR modes
  - Generate sufficient keystream
  - Encrypt the message and
  - Throw away keystream that is not needed
- For CFB, CBC, and ECB modes we might need to pad some input block to a multiple of s bits in the case of CFB mode and a multiple of b bits in the cases of CBC and ECB.
- Variety of padding methods have been proposed
- Most popular $10^{\star}$
- Many attacks reported due to inapt padding
- Padding Oracle Attack (Home Work)