(Applied) Cryptography

Week #3: Block Ciphers

Manuel Barbosa, mbb@fc.up.pt

MSI/MCC/MERSI – 2022/2023

DCC-FCUP

Part #1: Block Ciphers

What is a Block Cipher?

A block cipher is defined by two *deterministic* algorithms:

- Encipher $\mathbf{E}(K, P)$:
  - takes a key $K \in \{0, 1\}^{\lambda}$
  - takes a plaintext block $P \in \{0, 1\}^{B}$
  - outputs a ciphertext block $C \in \{0, 1\}^{B}$

What is a Block Cipher?

A block cipher is defined by two *deterministic* algorithms:

- Encipher $\mathbf{E}(K, P)$:
  - takes a key $K \in \{0,1\}^\lambda$
  - takes a plaintext block $P \in \{0,1\}^B$
  - outputs a ciphertext block $C \in \{0,1\}^B$
- Decipher $\mathbf{D}(K, C)$:
  - takes a key $K \in \{0,1\}^\lambda$
  - takes a ciphertext block $C \in \{0,1\}^B$
  - outputs a plaintext block $P \in \{0,1\}^B$

What is a Block Cipher?

A block cipher is defined by two *deterministic* algorithms:

- Encipher $\mathbf{E}(K, P)$:
  - takes a key $K \in \{0,1\}^\lambda$
  - takes a plaintext block $P \in \{0,1\}^B$
  - outputs a ciphertext block $C \in \{0,1\}^B$
- Decipher $\mathbf{D}(K, C)$:
  - takes a key $K \in \{0,1\}^\lambda$
  - takes a ciphertext block $C \in \{0,1\}^B$
  - outputs a plaintext block $P \in \{0,1\}^B$

A block cipher is **invertible**: each K defines a **permutation**.

What does security mean for block ciphers?

Block cipher should be a **pseudorandom permutation** (PRP).

What does security mean for block ciphers?

Block cipher should be a **pseudorandom permutation** (PRP).

We define this using an experiment:

- Experiment samples uniformly at random:
  - $K \in \{0,1\}^k$
  - permutation $\pi : \{0,1\}^B \to \{0,1\}^B$
  - bit b

What does security mean for block ciphers?

Block cipher should be a **pseudorandom permutation** (PRP).

We define this using an experiment:

- Experiment samples uniformly at random:
  - $K \in \{0,1\}^k$
  - permutation $\pi : \{0,1\}^B \to \{0,1\}^B$
  - bit b
- Attacker can (adaptively) ask for encryptions:
  - Attacker queries P
  - If $b = 0$ experiment returns $\mathbf{E}(K, P)$
  - If $b = 1$ experiment returns $\pi(P)$

What does security mean for block ciphers?

Block cipher should be a **pseudorandom permutation** (PRP).

We define this using an experiment:

- Experiment samples uniformly at random:
  - $K \in \{0,1\}^k$
  - permutation $\pi : \{0,1\}^B \to \{0,1\}^B$
  - bit b
- Attacker can (adaptively) ask for encryptions:
  - Attacker queries P
  - If $b = 0$ experiment returns $\mathbf{E}(K, P)$
  - If $b = 1$ experiment returns $\pi(P)$
- Attacker eventually returns b'

What does security mean for block ciphers?

Block cipher should be a **pseudorandom permutation** (PRP).

We define this using an experiment:

- Experiment samples uniformly at random:
  - $K \in \{0,1\}^k$
  - permutation $\pi : \{0,1\}^B \rightarrow \{0,1\}^B$
  - bit b
- Attacker can (adaptively) ask for encryptions:
  - Attacker queries P
  - If $b = 0$ experiment returns $\mathbf{E}(K, P)$
  - If $b = 1$ experiment returns $\pi(P)$
- Attacker eventually returns b'

Advantage: $\epsilon := |\Pr[\, b = b' \,] - 1/2\,|$

Implications of PRP security

What is a random permutation $\pi : \{0,1\}^B \rightarrow \{0,1\}^B$?

- Huge table with 2^B entries, indexed by P
- Each entry contains C
- Each C is sampled uniformly at random without repeats

Implications of PRP security

What is a random permutation $\pi : \{0,1\}^B \to \{0,1\}^B$?

- Huge table with 2^B entries, indexed by P
- Each entry contains C
- Each C is sampled uniformly at random without repeats

Difference to purely random function: no repeats.

Implications of PRP security

What is a random permutation $\pi : \{0,1\}^B \to \{0,1\}^B$?

- Huge table with 2^B entries, indexed by P
- Each entry contains C
- Each C is sampled uniformly at random without repeats

Difference to purely random function: no repeats.

Implications:

- Ciphertext blocks look totally random
- Different inputs $\Rightarrow$ independent outputs
- Must be impossible to recover key:
  - otherwise one could check $C = \mathbf{E}(K, P)$

Variants of PRP security

Strong PRPs:

- Attacker can also ask for block cipher decryptions
- How must experiment be re-defined?

Variants of PRP security

Strong PRPs:

- Attacker can also ask for block cipher decryptions
- How must experiment be re-defined?

Tweakable block-ciphers:

- Block cipher takes an extra argument T (a tweak)
- Both in encryption and in decryption
- Must behave as an independent permutation for all T
- How must experiment be re-defined?

Variants of PRP security

Strong PRPs:

- Attacker can also ask for block cipher decryptions
- How must experiment be re-defined?

Tweakable block-ciphers:

- Block cipher takes an extra argument T (a tweak)
- Both in encryption and in decryption
- Must behave as an independent permutation for all T
- How must experiment be re-defined?

Tweakable block ciphers can be constructed from block ciphers.

Block size

The Data Encryption Standard (70s-90s): $B = 64$.

The Advanced Encryption Standard (2000s-): $B = 128$.

Block must be small: efficient HW/SW implementation.

Block size

The Data Encryption Standard (70s-90s): $B = 64$.

The Advanced Encryption Standard (2000s-): $B = 128$.

Block must be small: efficient HW/SW implementation.

Block cannot be too small:

- Constructions based on block ciphers
- Key space 2^λ must be large
- Block size must be $B \sim \lambda$

E.g., some encryption schemes based on block ciphers constructions are insecure if block size is too small (64 could be problematic).

See this link for research on this.

Part #2: How are block ciphers built?

Iterated ciphers: rounds

Shorter descriptions and code/HW footprints:

- Simple and efficient round algorithm **R**
- Round algorithm is not secure as a block cipher
- Block cipher iterates round algorithm n times

Iterated ciphers: rounds

Shorter descriptions and code/HW footprints:

- Simple and efficient round algorithm **R**
- Round algorithm is not secure as a block cipher
- Block cipher iterates round algorithm *n* times
- Each round takes a different key:
    - Round key derived from block cipher key
    - Sequence of round keys called *key schedule*

Iterated ciphers: rounds

Shorter descriptions and code/HW footprints:

- Simple and efficient round algorithm **R**
- Round algorithm is not secure as a block cipher
- Block cipher iterates round algorithm *n* times

- Each round takes a different key:
  - Round key derived from block cipher key
  - Sequence of round keys called *key schedule*

- Deciphering has typically the same structure

$$\mathbf{E}(K,P) := \mathbf{R}(\ldots\mathbf{R}(\mathbf{R}(P,K_1),K_2)\ldots,K_n)$$
$$\mathbf{D}(K,C) := \mathbf{R}^{-1}(\ldots\mathbf{R}^{-1}(\mathbf{R}^{-1}(C,K_n),K_{n-1})\ldots,K_1)$$

Substitution-Permutation Networks (SPN)

The round function is a Substitution-Permutation layer.

- **Substitution** - S-boxes are small lookup tables (4-8 bits) designed to introduce non-linearity in the round function. They create *confusion*.
- **Permutation** - Bit-level transformations (e.g. switches) or algebraic functions that introduce dependencies across the whole block (diffusion).

Substitution-Permutation Networks (SPN)

The round function is a Substitution-Permutation layer.

- **Substitution** - S-boxes are small lookup tables (4-8 bits) designed to introduce non-linearity in the round function. They create *confusion*.
- **Permutation** - Bit-level transformations (e.g. switches) or algebraic functions that introduce dependencies across the whole block (diffusion).

Both need to be efficient in HW/SW.

Substitution-Permutation Networks (SPN)

The round function is a Substitution-Permutation layer.

- **Substitution** - S-boxes are small lookup tables (4-8 bits) designed to introduce non-linearity in the round function. They create *confusion*.
- **Permutation** - Bit-level transformations (e.g. switches) or algebraic functions that introduce dependencies across the whole block (diffusion).

Both need to be efficient in HW/SW.

S-boxes heuristically designed to:

- Create complex relation between input/output
- Minimize statistical bias in outputs

Substitution-Permutation Networks (SPN)

The round function is a Substitution-Permutation layer.

- **Substitution** - S-boxes are small lookup tables (4-8 bits) designed to introduce non-linearity in the round function. They create *confusion*.
- **Permutation** - Bit-level transformations (e.g. switches) or algebraic functions that introduce dependencies across the whole block (diffusion).

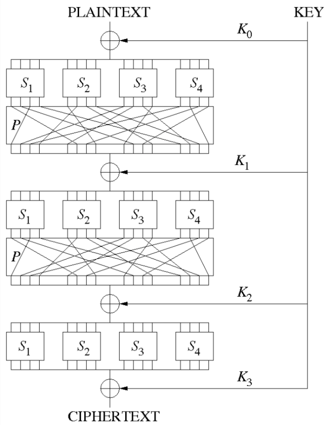
Both need to be efficient in HW/SW.

S-boxes heuristically designed to:

- Create complex relation between input/output
- Minimize statistical bias in outputs

Example block cipher: AES

Substitution-Permutation Networks (2)



(from Wikipedia)

Feistel Networks

The round function only processes half of the block:

- Input block is seen as pair (L, R)
- Output block is $(R \oplus F(K_i, L), L)$
- F is called the round function

Unprocessed half-block is masked on the next round.

Note that decryption is identical to encryption:

- Only key schedule is inverted
- Hugely important in the 70s for HW implementation

Feistel Networks

The round function only processes half of the block:

- Input block is seen as pair (L, R)
- Output block is $(R \oplus F(K_i, L), L)$
- F is called the round function

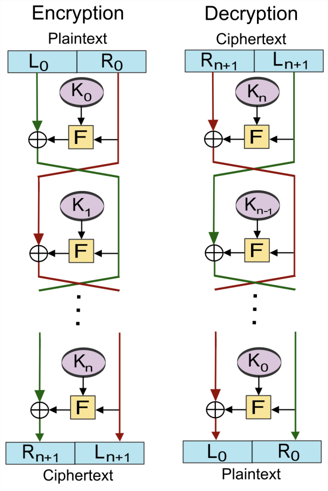
Unprocessed half-block is masked on the next round.

Note that decryption is identical to encryption:

- Only key schedule is inverted
- Hugely important in the 70s for HW implementation

Example block cipher: DES, GOST

Feistel Networks (2)



Feistel Networks (3)

Round function can be a PRP or a PRF:

- A PRF is similar to a PRP but not necessarily invertible
- Input size can be different from output size
- Security experiment is similar to PRP:
  - Experiment chooses random function f
  - Rather than random permutation π

Feistel Networks (3)

Round function can be a PRP or a PRF:

- A PRF is similar to a PRP but not necessarily invertible
- Input size can be different from output size
- Security experiment is similar to PRP:
  - Experiment chooses random function f
  - Rather than random permutation π

Strong theoretical results if round function is ideal:

- 4 rounds are enough for strong PRP!

Feistel Networks (3)

Round function can be a PRP or a PRF:

- A PRF is similar to a PRP but not necessarily invertible
- Input size can be different from output size
- Security experiment is similar to PRP:
  - Experiment chooses random function f
  - Rather than random permutation π

Strong theoretical results if round function is ideal:

- 4 rounds are enough for strong PRP!

Practical block ciphers use extra rounds:

- round functions are heuristically designed

Advanced Encryption Standard (AES)

AES was standardized in 2000:

- DES was still the standard (56-bit keys!)
- 3DES was a common solution for short keys (112-bit security)
- 3DES: use DES 3 times (EDE) with 3 independent keys
- Still short block

Advanced Encryption Standard (AES)

AES was standardized in 2000:

- DES was still the standard (56-bit keys!)
- 3DES was a common solution for short keys (112-bit security)
- 3DES: use DES 3 times (EDE) with 3 independent keys
- Still short block

AES is now the most used block cipher, by far.

- available in mainstream CPUs as HW implementation.

Advanced Encryption Standard (AES)

AES was standardized in 2000:

- DES was still the standard (56-bit keys!)
- 3DES was a common solution for short keys (112-bit security)
- 3DES: use DES 3 times (EDE) with 3 independent keys
- Still short block

AES is now the most used block cipher, by far.

- available in mainstream CPUs as HW implementation.

AES was selected as a result of a competition:

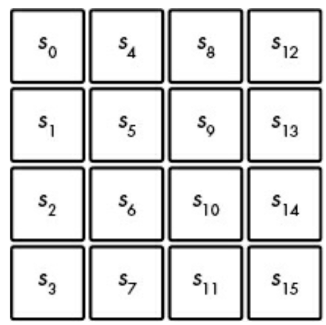
- 1997-2000 public competition run by NIST
- This process has since become the norm
- Open to proposals, scrutinized by community
- Criteria: performance and resistance to cryptanalysis

AES internals

Block-size 128-bits and varying key size (**128**, 192, 256)-bits.

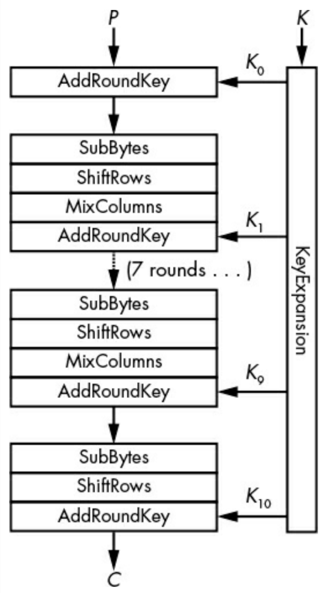
Keeps a 128-bit internal state: 4×4 array of 16-bytes.

State is transformed using a substitution-permutation network.



Substitutions/permutations have an algebraic description.

AES internals (2)



AES internals (3)

The substitution-permutation network uses:

- **AddRoundKey** - Full XOR with the state
- **SubBytes** - Replace each byte using lookup table (S-box)
- **ShiftRows** - Matrix rows are shifted 0..3 positions.
- **MixColumns** - Columns linearly transformed

AES internals (3)

The substitution-permutation network uses:

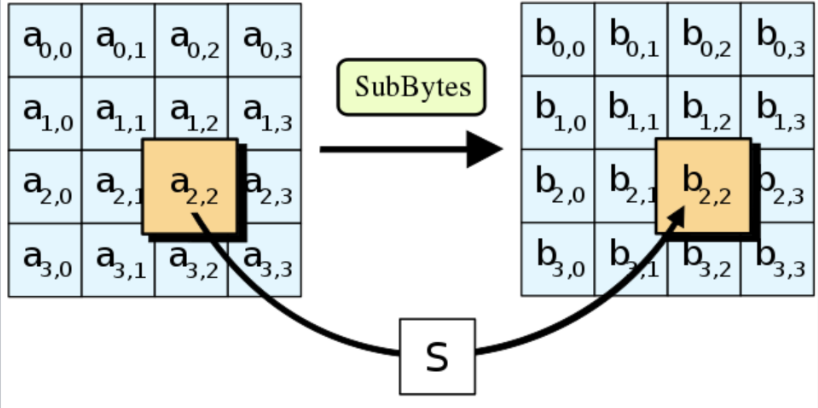
- **AddRoundKey** - Full XOR with the state
- **SubBytes** - Replace each byte using lookup table (S-box)
- **ShiftRows** - Matrix rows are shifted 0..3 positions.
- **MixColumns** - Columns linearly transformed

SubBytes performs the substitution part.

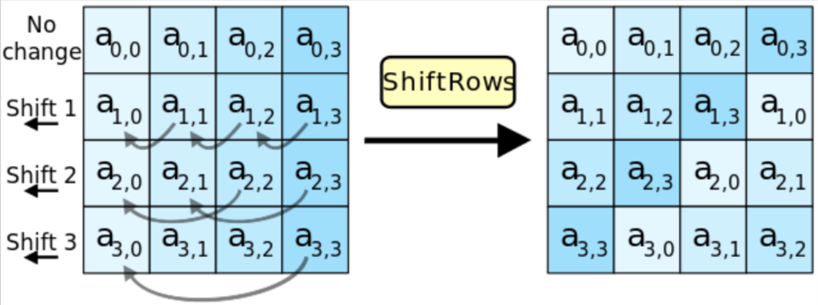
ShiftRows and **MixColumns** are the permutation.

Last round does not **MixColumns**. Why? (see **here**)

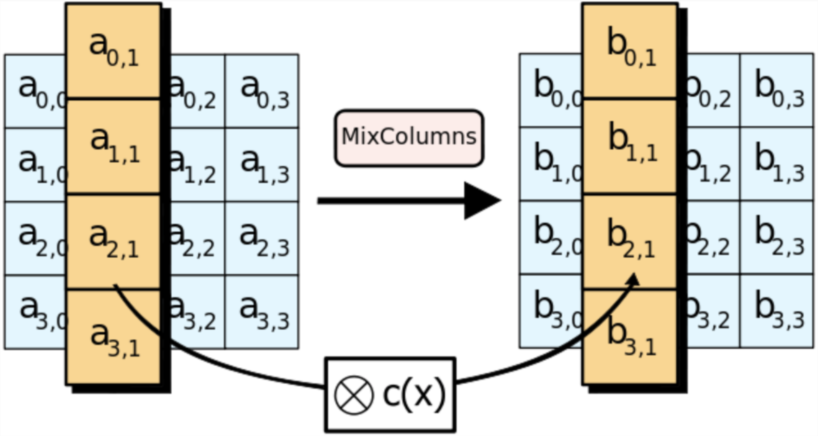
AES internals: SubBytes (Wikipedia)



AES internals: ShiftRows (Wikipedia)



AES internals: MixColumns (Wikipedia)



AES Implementations

AES is hard to efficiently implement in software:

- Naive implementations using tables leak via side-channels
- Removing side-channels in software is hard

But . . .

AES Implementations

AES is hard to efficiently implement in software:

- Naive implementations using tables leak via side-channels
- Removing side-channels in software is hard

But . . .

AES is super-fast in mainstream processors:

- **AES-NI** - AES Native Instructions
- From SW one can use HW AES

Is AES Secure?

There is no mathematical proof that AES is a PRP.

All practical applications based on AES **assume** this.

Why?

Is AES Secure?

There is no mathematical proof that AES is a PRP.

All practical applications based on AES **assume** this.

Why?

AES has been around for almost 25 years:

- No significant cryptanalysis progress
- AES scrutiny is an important area of research
- Direct attack on AES unlikely to be the weakest link

Is AES Secure?

There is no mathematical proof that AES is a PRP.

All practical applications based on AES **assume** this.

Why?

AES has been around for almost 25 years:

- No significant cryptanalysis progress
- AES scrutiny is an important area of research
- Direct attack on AES unlikely to be the weakest link

Assuming AES is a PRP we have provably secure and very efficient symmetric encryption.

Part #3: Symmetric Encryption from Block Ciphers

So-called modes of operation

Historically, block-ciphers were used in different modes of operation to encrypt data.

Modern cryptography clarifies things:

- Block-ciphers are a **primitive**
- On their own they are useless
- There are **totally insecure ways** to encrypt with a block cipher
- Encryption schemes have their own security definitions
- We build secure encryption schemes from block ciphers
- We prove encryption secure assuming block cipher PRP

Symmetric Encryption

Syntax:

- Key Generation: Typically uniform sampling in $\{0, 1\}^{\lambda}$
- Encryption: Probabilistic algorithm $C \twoheadleftarrow \textbf{Enc}(K, M)$
- Decryption: Deterministic algorithm $M / \perp \leftarrow \textbf{Dec}(K, C)$

Symmetric Encryption

Syntax:

- Key Generation: Typically uniform sampling in $\{0,1\}^\lambda$
- Encryption: Probabilistic algorithm $C \twoheadleftarrow \mathbf{Enc}(K, M)$
- Decryption: Deterministic algorithm $M/\perp \leftarrow \mathbf{Dec}(K, C)$

Security (IND-CPA):

- Experiment samples K and bit b uniformly at random
- Attacker can (adaptively) get encryptions chosen messages
- Attacker outputs (M_0, M_1) s.t. $|M_0| = |M_1|$
- Attacker gets $C^* \twoheadleftarrow \mathbf{Enc}(K, M_b)$
- Attacker can (adaptively) get encryptions chosen messages

Attacker eventually returns b'

Symmetric Encryption

Syntax:

- Key Generation: Typically uniform sampling in $\{0,1\}^\lambda$
- Encryption: Probabilistic algorithm $C \twoheadleftarrow \mathbf{Enc}(K, M)$
- Decryption: Deterministic algorithm $M/\perp \leftarrow \mathbf{Dec}(K, C)$

Security (IND-CPA):

- Experiment samples K and bit b uniformly at random
- Attacker can (adaptively) get encryptions chosen messages
- Attacker outputs (M_0, M_1) s.t. $|M_0| = |M_1|$
- Attacker gets $C^* \twoheadleftarrow \mathbf{Enc}(K, M_b)$
- Attacker can (adaptively) get encryptions chosen messages

Attacker eventually returns b'

Advantage: $\epsilon := |\Pr[\, b = b' \,] - 1/2\,|$

Insecure encryption with a block cipher

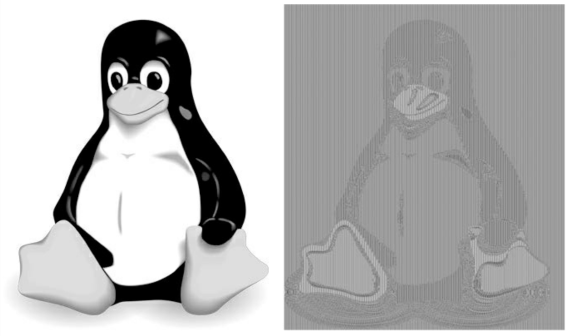
Electronic-Code-Book mode:

- Break message into plaintext blocks P_i
- Last block may need padding (more on padding later)
- Independently encipher each block $C_i \leftarrow \mathbf{E}(K, P_i)$

Insecure encryption with a block cipher

Electronic-Code-Book mode:

- Break message into plaintext blocks P_i
- Last block may need padding (more on padding later)
- Independently encipher each block $C_i \leftarrow \mathbf{E}(K, P_i)$



Insecure encryption with a block cipher (2)

What is the problem?

- Equal input blocks $\Rightarrow$ Equal output blocks
- Preserves patterns that vary slower than block size

What happens in the security experiment?

Insecure encryption with a block cipher (2)

What is the problem?

- Equal input blocks $\Rightarrow$ Equal output blocks
- Preserves patterns that vary slower than block size

What happens in the security experiment?

Here's an attacker that always wins the experiment:

- Output $M_0 \neq M_1$, where $|M_0| = |M_1|$
- Ask for an encryption of M_0 to get C
- Return $b' = 0$ iff $C^* = C$

Attack works against all deterministic encryption schemes.

Insecure encryption with a block cipher (2)

What is the problem?

- Equal input blocks $\Rightarrow$ Equal output blocks
- Preserves patterns that vary slower than block size

What happens in the security experiment?

Here's an attacker that always wins the experiment:

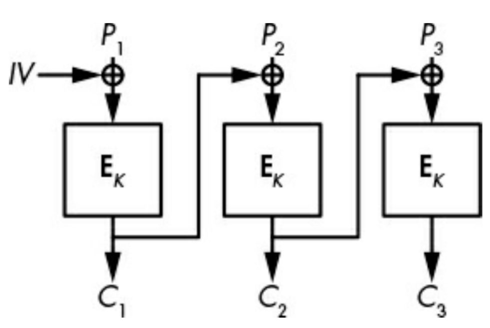
- Output $M_0 \neq M_1$, where $|M_0| = |M_1|$
- Ask for an encryption of M_0 to get C
- Return $b' = 0$ iff $C^* = C$

Attack works against all deterministic encryption schemes.

Real-world example of this attack?

Cipher Block Chaining (CBC)

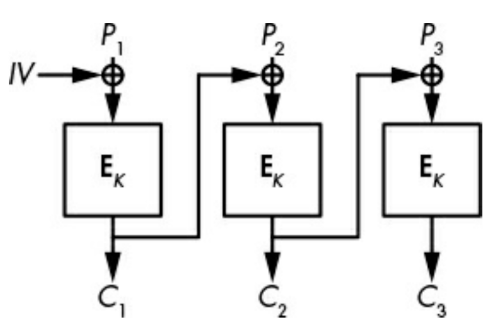
Engineers designed a secure encryption scheme before security proofs were well understood.



What is the main difference to ECB?

Cipher Block Chaining (CBC)

Engineers designed a secure encryption scheme before security proofs were well understood.



What is the main difference to ECB?

Random block-size initialization vector (IV).

Cipher Block Chaining (CBC): Security/Performance

Intuition of CBC security:

- Random IV makes first block-cipher input random
- Block cipher security implies C_1 looks random and indendent of everything else
- CBC uses C_1 as IV for remaining ciphertexts
- Use the same argument for C_2, etc.
- Two encryptions of same plaintext look independent

Cipher Block Chaining (CBC): Security/Performance

Intuition of CBC security:

- Random IV makes first block-cipher input random
- Block cipher security implies C_1 looks random and indendent of everything else
- CBC uses C_1 as IV for remaining ciphertexts
- Use the same argument for C_2, etc.
- Two encryptions of same plaintext look independent

How does decryption work?

Cipher Block Chaining (CBC): Security/Performance

Intuition of CBC security:

- Random IV makes first block-cipher input random
- Block cipher security implies C_1 looks random and indendent of everything else
- CBC uses C_1 as IV for remaining ciphertexts
- Use the same argument for C_2, etc.
- Two encryptions of same plaintext look independent

How does decryption work?

Suppose large encrypted file:

- How can you decrypt arbitrary block? Parallelism?
- How can you modify encryption of plaintext block?

Cipher Block Chaining (CBC): Padding

There are several padding methods:

- Some schemes require message size to be multiple of block size
- Padding schemes re-encode message so that this is true
- To avoid ambiguity: **padding is always added**.

Cipher Block Chaining (CBC): Padding

There are several padding methods:

- Some schemes require message size to be multiple of block size
- Padding schemes re-encode message so that this is true
- To avoid ambiguity: **padding is always added**.

Most common padding scheme is specified in PKCS #7:

- Let $k > |M|$ be the next multiple of B (in bytes)
- Add $k - |M|$ bytes with the value $k - |M| + 1$

How to decode?

Cipher Block Chaining (CBC): Padding

There are several padding methods:

- Some schemes require message size to be multiple of block size
- Padding schemes re-encode message so that this is true
- To avoid ambiguity: **padding is always added**.

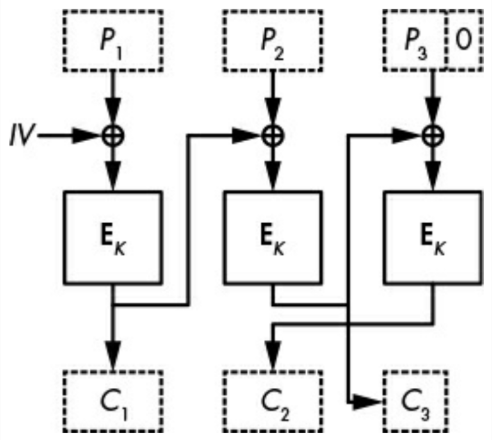
Most common padding scheme is specified in PKCS #7:

- Let $k > |M|$ be the next multiple of B (in bytes)
- Add $k - |M|$ bytes with the value $k - |M| + 1$

How to decode?

Overhead is at least one byte and at most one block.

Alternative to CBC padding: Ciphertext Stealing



Not widely used.

Counter Mode (CTR)

Progress in provable security $\Rightarrow$ simplest mode of operation:

- generate random block-size counter ctr
- generate key stream of sufficient size:

$$\mathbf{E}(K, \mathrm{ctr}) \| \mathbf{E}(K, \mathrm{ctr} + 1) \| \ldots \| \mathbf{E}(K, \mathrm{ctr} + k)$$

- XOR plaintext and (truncated) key stream
- Ciphertext also includes counter (why?)

Counter Mode (CTR)

Progress in provable security $\Rightarrow$ simplest mode of operation:

- generate random block-size counter ctr
- generate key stream of sufficient size:

$$\mathbf{E}(K, \mathrm{ctr}) \| \mathbf{E}(K, \mathrm{ctr}+1) \| \ldots \| \mathbf{E}(K, \mathrm{ctr}+k)$$

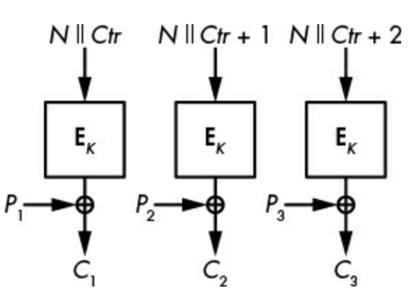
- XOR plaintext and (truncated) key stream
- Ciphertext also includes counter (why?)

Security intuition:

- Let us assume counters never repeat (how likely?)
- PRP security guarantees key-stream looks random
- CTR mode is essentially a One-Time-Pad approximation

Nonce-based encryption

Often Counter Mode is used in Nonce-Based form:



Encryptor guarantees unique N:

- Encryption becomes stateful
- Security experiment is changed (how?)

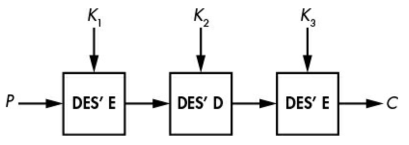
Counter Mode Performance

Counter mode is very efficient:

- Key stream can be pre-processed
- Any part of the data can be accessed efficiently
- This includes read/write access
- Decryption/encryption can be parallelized

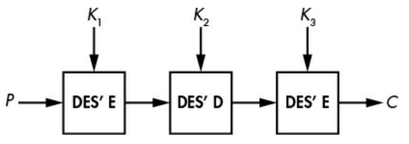
For these reasons, many modern protocols rely on CTR mode.

What can go wrong in block-cipher design

3DES uses three DES instances: 

One would hope for $3 * 56 = 168$-bit security.

What can go wrong in block-cipher design

3DES uses three DES instances: 

One would hope for $3 * 56 = 168$-bit security.

Meet-in-the-Middle attacks mean we only get 112-bit security:

- Given (P, C) we find the key as follows
- Construct 2^{56} table of $\mathbf{D}(K, C)$ for all K
- Try all (K_1, K_2) enciphering P and check in table
- Overall $2^{112} + 2^{56} \approx 2^{112}$ work (memory?)

What can go wrong in modes of operation

What does IND-CPA model say?

- Attacker has access to encryptions
- Can't extract any information about messages
- What if it has access to side information on decryption?
- No guarantee modified ciphertext is rejected: what leaks?

What can go wrong in modes of operation

What does IND-CPA model say?

- Attacker has access to encryptions
- Can't extract any information about messages
- What if it has access to side information on decryption?
- No guarantee modified ciphertext is rejected: what leaks?

Practical example:

- Padding oracle attacks against CBC (TLS 1.*)
- Attacker gets to observe padding check error
- This is enough to recover plaintext (e.g., cookies)

What can go wrong in modes of operation

What does IND-CPA model say?

- Attacker has access to encryptions
- Can't extract any information about messages
- What if it has access to side information on decryption?
- No guarantee modified ciphertext is rejected: what leaks?

Practical example:

- Padding oracle attacks against CBC (TLS 1.*)
- Attacker gets to observe padding check error
- This is enough to recover plaintext (e.g., cookies)

Root problem: processing non-authenticated ciphertext.

Thank you!

mbb@fc.up.pt

http://www.dcc.fc.up.pt/~mbb