Block Ciphers

Chanathip Namprempre

Computer Science
Reed College

Agenda: Block ciphers

1. General idea
2. DES
3. AES
4. Modelling block ciphers
5. Attacks against block ciphers
6. Security notions for block ciphers (PRP)
7. Related security notion (PRF)
8. Example security analysis

Block ciphers: General idea

Properties

- Given M and K, it's easy to compute C.
- Given C and K, it's easy to compute M.
- With C but without K, it should be hard to compute M.
- In fact, with C but without K, it should be hard to compute any partial information about M. (e.g. first bit, last bit, parity, in English?, etc.)

Block Ciphers

BLOCK CIPHERS are the main tool for doing symmetric-key cryptography.

If we use it well, we'll get something good.
Otherwise, we won't even if the block cipher is excellent!

Our focus: how to use it.
(How to design it is still kind of an *art*.)

Data Encryption Standard (DES)

DES: History

Every time you use the ATM , you're most likely using DES!

History

1972:	NBS(now NIST) asks for something for encryption.
1974:	IBM replied with Lucifer algorithm $\Rightarrow$ became DES.
later:	ANSI, American Bankers Assoc. adopted DES.
	Recertified every 5 years till AES.

DES: Key length $k = 56$, Block length $n = 64$

$\forall K \in \{0,1\}^{56}, \mathrm{DES}_K(\cdot)$ is a permutation.

DES is very fast in hardware!

DES: forward direction

DES Algorithm

Algorithm $\text{DES}_K(M)$

$(K_1, \ldots, K_{16}) \leftarrow \text{KeySchedule}(K)$ // $|K_i| = 48$ for all $1 \le i \le 16$

$M \leftarrow IP(M)$

Parse M as $L_0 \| R_0$ // $|L_0| = |R_0| = 32$

for $r = 1$ to 16 do

$L_r \leftarrow R_{r-1}$

$R_r \leftarrow f(K_r, R_{r-1}) \oplus L_{r-1}$

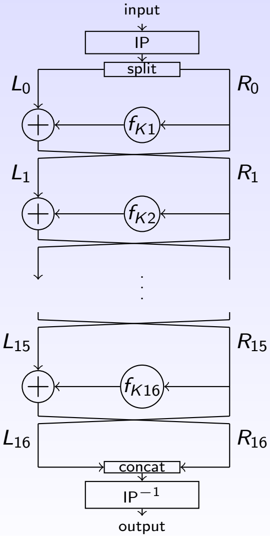
endfor

$C \leftarrow IP^{-1}(L_{16} \| R_{16})$

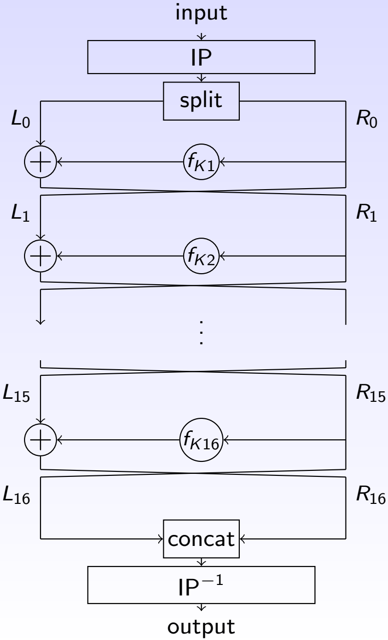
return C

DES: forward direction

Algorithm $\text{DES}_K(M)$
    $(K_1, \ldots, K_{16}) \leftarrow \text{KeySchedule}(K)$
    $M \leftarrow IP(M)$
    Parse M as $L_0 \| R_0$ // $|L_0| = |R_0| = 32$
    for $r = 1$ to 16 do
        $L_r \leftarrow R_{r-1}$
        $R_r \leftarrow f(K_r, R_{r-1}) \oplus L_{r-1}$
    endfor
    $C \leftarrow IP^{-1}(L_{16} \| R_{16})$
    return C



DES: Pictorially



This structure is called a **Feistel Network**.

We need to look at

- IP, IP^{-1}
- $f : E, P, S$
- KeySchedule : PC-1, PC-2

Decryption uses the same circuit for f!

DES: IP, IP^{-1}

Just permute bits.

IP

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

IP^{-1}

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

How to read the tables:

- IP: 1st bit of output is the 58th bit of the input.
- IP^{-1}: 1st bit output is the 40th bit of the input.

DES: $f(K, R)$

function $f(K, R)$ // $|K| = 48$ and $|R| = 32$

$R \leftarrow E(R)$; $R \leftarrow R \oplus K$

Parse R as $R_1 \| R_2 \| R_3 \| R_4 \| R_5 \| R_6 \| R_7 \| R_8$ // $|R_i| = 6$ for $1 \leq i \leq 8$

for $i = 1, \ldots, 8$ do

$R_i \leftarrow S_i(R_i)$ // Each S-box returns 4 bits

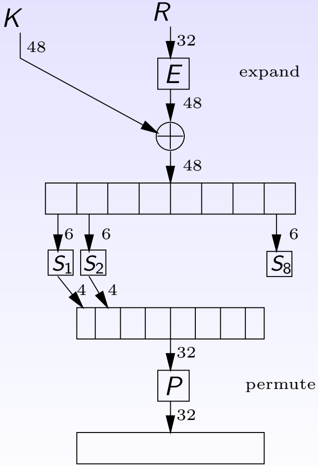
$R \leftarrow R_1 \| R_2 \| R_3 \| R_4 \| R_5 \| R_6 \| R_7 \| R_8$

$R \leftarrow P(R)$

Return R

DES: $f(K, R)$

input: 48-bit subkey K, 32-bit input R
output: 32-bit output R



E

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

P

16	7	20	21
29	12	28	17
1	15	23	26
5	18	31	10
2	8	24	14
32	27	3	9
19	13	30	6
22	11	4	25

Read the same way as IP.

DES: S-boxes

input: 6 bits $b_1 b_2 b_3 b_4 b_5 b_6$
output: 4 bits

Read row $b_1 b_2$ column $b_3 b_4 b_5 b_6$ to get output an integer in the range $0, \ldots, 15$.

For example, take S_1:

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
0	1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
1	0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
1	1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

There are 8 tables for $S_1, \ldots, S_8$.

DES: KeySchedule

inputs: key (56 bits)
outputs: 16 round keys (48 bits each)

function KeySchedule(K) // $|K| = 56$
 $K \leftarrow$ PC-1(K)
 Parse K as $C_0 \| D_0$
 For $r = 1, \ldots, 16$ do
  If $r \in \{1, 2, 9, 16\}$ then $j \leftarrow 1$ else $j \leftarrow 2$
  $C_r \leftarrow \text{leftshift}_j(C_{r-1}) D_r \leftarrow \text{leftshift}_j(D_{r-1})$
  $K_r \leftarrow$ PC-2($C_r \| D_r$)
 Return $(K_1, \ldots, K_{16})$

[For rounds # 1,2,9,16, left shift by 1 bit.
For all other rounds, left shift by 2 bits.]

DES: KeySchedule (cont.)

PC-2 is read as usual. PC-1 is a little more complicated.

PC-1 (permutes)

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

PC-2 (shrink)

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

How to read PC-1: Suppose input is $K[1] \ldots K[56]$ and output is $L[1] \ldots L[56]$. How to get $L[1]$?

1. The first entry of PC-1 is 57.
2. Write 57 in the form of $8q + r$. So $q = 7$ and $r = 1$.
3. $L[1] = K[57 - q] = K[57 - 7] = K[50]$.
4. So the first bit of the output is the 50th bit of the input.

AES

AES (Advanced Encryption Standard)

- AES is a special case of Rijndael i.e.
  block length $= 128$
  key length $= 128$ or 192 or 256
- More documented than DES with design rationales given.
- Vague security still (just like DES), i.e. it's good because we don't know how to break it.

See animation.

Modelling block ciphers

Modelling block ciphers: Pictorially

Modelling Block Ciphers

A block cipher = function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$

k = key length
n = block length

But we often look at E as a **family of functions** where each function maps n bits to n bits and is indexed by $K \in \{0,1\}^k$.

Modelling block ciphers: Pictorially

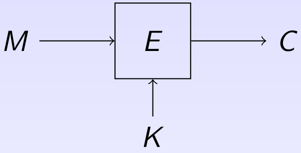
Modelling Block Ciphers

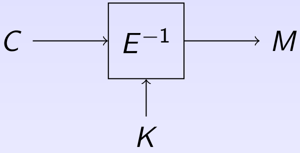
A block cipher = function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$

k = key length
n = block length

But we often look at E as a **family of functions** where each function maps n bits to n bits and is indexed by $K \in \{0,1\}^k$.

Modelling block ciphers: Pictorially





Modelling Block Ciphers

A block cipher = function $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$

k = key length
n = block length

But we often look at E as a **family of functions** where each function maps n bits to n bits and is indexed by $K \in \{0,1\}^k$.

Functions and permutations

What is a function?

- $f : A \to B$
- Every member of A must be mapped to exactly one member in B under f.

What is a permutation?

- $p : A \to B$
- p must be a function.
- p must be one-to-one and onto.

Pondering $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$

Try to think about what each of these objects look like, both as a function and a family of functions.

1. $\{0,1\}^1 \times \{0,1\}^1 \to \{0,1\}^1$
2. $\{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^1$
3. $\{0,1\}^2 \times \{0,1\}^1 \to \{0,1\}^2$
4. $\{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$
5. $\{0,1\}^2 \times \{0,1\}^3 \to \{0,1\}^2$
6. $\{0,1\}^3 \times \{0,1\}^2 \to \{0,1\}^2$

Modelling block Ciphers: notation

Notation:
Fix $K \in \{0,1\}^k$. Then,

- $E_K : \{0,1\}^n \to \{0,1\}^n$ where $E_K(M) = E(K, M)$
- $E_K^{-1} : \{0,1\}^n \to \{0,1\}^n$ is the inverse permutation of $E_K(\cdot)$
- We require that $\forall x \in \{0,1\}^n$,

$$E_K^{-1}(E_K(x)) = x \text{ and } E_K(E_K^{-1}(x)) = x$$

- $E^{-1} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ where, $\forall K \in \{0,1\}^k$,

$$E_K^{-1}(C) = E^{-1}(K, C)$$

Definition

E^{-1} is the inverse block cipher to E.

Using the notation: Example

Let $E : \{0,1\}^2 \times \{0,1\}^2 \to \{0,1\}^2$ be a family of permutations with the following maps:

$$\begin{array}{rl} E_{00}: & (10, 01, 11, 00) \\ E_{01}: & (01, 00, 11, 10) \\ E_{10}: & (00, 11, 01, 10) \\ E_{11}: & (01, 10, 00, 11) \end{array}$$

Questions

1. If $K = 11$ and you want to send the message 00, what ciphertext value should you send?
2. If $K = 10$ and you receive a ciphertext 11, what was the intended message?

Modelling Block Ciphers: desired properties

Note that

- E is **public** and fully specified.
- E, E^{-1} are **easily computable**.

  [There are public and efficient programs for this.]
  [i.e. given K, M, it is easy to find $C = E_K(M)$ and $M = E_K^{-1}(C)$.]

Modelling Block Ciphers: no talk of security yet

Notice:

- We've only talked about **what a block cipher is**.
- We have NOT said anything about **what properties a good block cipher must have**.

Example

Let $E_K(\cdot)$ be an identity function.
[i.e. $E_K(M) = M$ for all $M \in \{0,1\}^n$]

This is syntactically a block cipher, but obviously not a good one.

Attacks against block ciphers

Attack types

- **Known message attack**:
  Attacker gets q pairs of M and C.

  $$\boxed{\text{Attacker}} \quad \begin{array}{ll} \Longleftarrow & (M_1, C_1), \ldots, (M_q, C_q) \\ \Longrightarrow & T = ? \end{array}$$

- **Chosen message attack**:
  Attacker gets a black box to which it can submit M_i to get C_i.

  $$\boxed{\text{Attacker}} \quad \begin{array}{ll} \Longleftarrow & E_T \\ \Longrightarrow & T = ? \end{array}$$

Key recovery attacks against block ciphers

Let $q \geq 0$ be an integer parameter.

$$\boxed{\text{Attacker}} \quad \begin{array}{ll} \Longleftarrow & (M_1, C_1), \ldots, (M_q, C_q), E_T \\ \Longrightarrow & T = ? \end{array}$$

- K is consistent with the input-output examples $(M_1, C_1), ..., (M_q, C_q)$ if $\forall 1 \leq i \leq q, E_K(M_i) = C_i$
- Let $Cons_E(M_1, C_1), ..., (M_q, C_q)$ be the set of consistent keys for the input-output pairs.
- The adversary succeeds if she can find a key in this set (may have more than 1 element).

Exhaustive key search via known message attacks

Given $(M_1, C_1), \ldots, (M_q, C_q)$, the attacker can follow either of these two strategies:

$\boxed{\text{Strategy 1}}$

Try $T_i \in \{0,1\}^k$ until $E_{T_i}(M_1) = C_1$.

Algorithm $\mathrm{EKS1}_{\mathrm{E}}(M_1, C_1)$
  for $i = 1, \ldots, 2^k$ do
    if $E(T_i, M_1) = C_1$ then return T_i

Notice

We can always do this! So **no block cipher is perfectly secure!**

Notice: This strategy could get us a wrong key.

Exhaustive key search via known message attacks (cont.)

Given $(M_1, C_1), \ldots, (M_q, C_q)$, the attacker can follow either of these two strategies:

$\boxed{\text{Strategy 2}}$

Try $T_i \in \{0,1\}^k$ until $E_{T_i}(M_1) = C_1$ and $\ldots$ and $E_{T_i}(M_q) = C_q$.

Algorithm $\text{EKS2}_{\text{E}}(M_1, C_1)$
for $i = 1, \ldots, 2^k$ do
if $E(T_i, M_1) = C_1 \wedge \ldots \wedge E(T_i, M_q) = C_q$
then return T_i

For DES, $q = 2$ is enough.

Runing time of exhaustive key search via known message attack

Suppose # of(M, C) pairs $= q = 1$

worst case: # of block cipher applications $= 2^k$

average case:
Let i be the random variable for # of block cipher application.

$$
\begin{aligned}
E[i] &= \sum_{i=1}^{2^k} i \cdot \Pr[\, K = T_i \,] \\
&= \sum_{i=1}^{2^k} \frac{i}{2^k} = \frac{1}{2^k} \sum_{i=1}^{2^k} i = \frac{1}{2^k} \frac{2^k \cdot (2^k + 1)}{2} \\
&\approx \frac{2^k}{2} = 2^{k-1} \, .
\end{aligned}
$$

DES challenge

Given $M_1 M_2 M_3$, C_1, C_2, C_3, C_4, C_5, C_6,
find K and compute M_4, M_5, M_6.

Summary from Dan Boneh:

- 1997 Internet search: 3 months
- 1998 EFF's Deep crack: 3 days (USD 250,000)
- 1999 Combined search: 22 hours
- 2006 COPACOBANA (120 FPGAs): 7 days (USD 10,000)

Exhaustive key search does not look inside DES.
It's a black-box attack.

Cryptanalysis of DES

Differential crytanalysis:	$q = 2^{47}$ find key with chosen-msg attack
Linear cryptanalysis:	$q = 2^{44}$ find key with known-msg attack

Practical? No. Require too many sample (M, C) pairs.
i.e. $2^{44} * 64 * 64 = 2.81 * 10^4$ bits $\approx$ 281 terabytes

Linear & differential cryptanalysis decimated other ciphers but not DES!

Band-aid

We want longer keys. Alternatives are

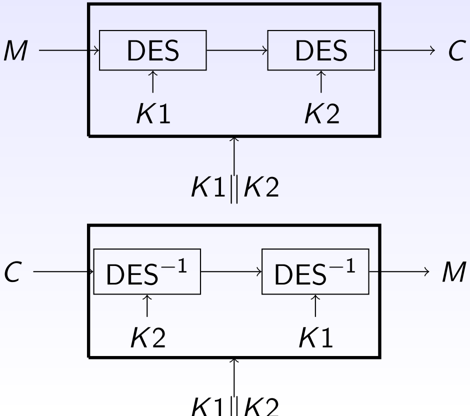
- Iterated DES : 2DES, 3DES3, 3DES2
- DESX

But we also want bigger blocks! $\Longrightarrow$ AES

Double DES: $2DES_{K1\|K2}(M) = DES_{K2}(DES_{K1}(M))$

Effective key length $= 57 < 112$

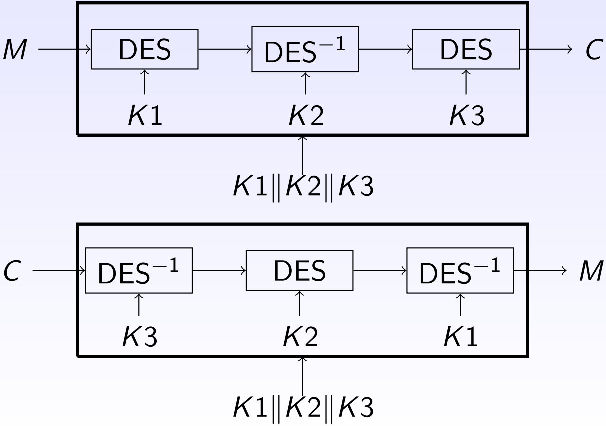
[Meet-in-the-middle, known-message attack: Make two lists $DES(K1_i, M)$ and $DES^{-1}(K2_j, C)$ and look for $K1, K2$ such that the values are equal.]



Try writing down the algorithm to perform the meet-in-the-middle attack and analyze the worst-case running time.

Triple DES: $3\text{DES}_{K1\|K2\|K3}(M) = \text{DES}_{K3}(\text{DES}^{-1}_{K2}(\text{DES}_{K1}(M)))$

Effective key length $= 112 < 168$ due to a meet-in-the-middle attack



Security notions for block ciphers

Key-recovery security

Key-recovery security is NOT enough.

- Security against key recovery is necessary
  Without it : given C, just compute $E_K^{-1}(C) = M$.
- But security against key recovery is not sufficient!

Consider a block cipher that reveals message bits without revealing key bits.

Clearly, we can't use it to encrypt things!

Key-recovery security

Example

Define $E : \{0,1\}^{128} \times \{0,1\}^{256} \to \{0,1\}^{256}$ as
$\forall K \in \{0,1\}^{128}, \forall M[1]M[2] \in \{0,1\}^{256}$ where $|M[1]| = |M[2]|$,

$$E_K(M) = \mathrm{AES}_K(M[1]) || M[2]$$

Key recovery is still hard [due to AES].
But half the message is revealed!

PRP

Q : So if what we need isn't hardness against key recovery, then what?

A : PRP

PRP-CPA

Definition (PRP-CPA)

Let k, n be positive integers, and let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a family of permutations. Let A be an adversary with access to an oracle. We define the following subroutines, experiment, and advantage function.

Subroutines

Subroutine *Initialize*
$b \xleftarrow{\$} \{0,1\}$
If $b = 1$ then $K \xleftarrow{\$} \{0,1\}^k$ else $p \xleftarrow{\$} \mathrm{Perm}(n)$

Subroutine $g(x)$
If $b = 1$ then return $E_K(x)$ else return $p(x)$

Subroutine *Finalize*(d)
Return (d = b)

Experiment

Experiment $\mathbf{Exp}_E^{\mathrm{prp\text{-}cpa}}(A)$

Initialize
$d \xleftarrow{\$} A^g$
Return *Finalize*(d)

We define the **prp-cpa advantage** of an adversary A mounting a chosen-plaintext attack against E as

$$\mathbf{Adv}_E^{\mathrm{prp\text{-}cpa}}(A) = 2 \cdot \Pr\left[\mathbf{Exp}_E^{\mathrm{prp\text{-}cpa}}(A) \Rightarrow \mathsf{true}\right] - 1 .$$

What does Perm(2) look like?

Perm(2) = set of all permutations mapping 2 bits to 2 bits.

(00,01,10,11)	(01,00,10,11)	(10,00,01,11)	(11,00,01,10)
(00,01,11,10)	(01,00,11,10)	(10,00,11,01)	(11,00,10,01)
(00,10,01,11)	(01,10,00,11)	(10,01,00,11)	(11,01,00,10)
(00,10,11,01)	(01,10,11,00)	(10,01,11,00)	(11,01,10,00)
(00,11,01,10)	(01,11,00,10)	(10,11,00,01)	(11,10,00,01)
(00,11,10,01)	(01,11,10,00)	(10,11,01,00)	(11,10,01,00)

What does Func(2, 1) look like?

Func(2, 1) = set of all functions mapping 2 bits to 1 bit.

(0,0,0,0)	(0,0,0,1)	(0,0,1,0)	(0,0,1,1)
(0,1,0,0)	(0,1,0,1)	(0,1,1,0)	(0,1,1,1)
(1,0,0,0)	(1,0,0,1)	(1,0,1,0)	(1,0,1,1)
(1,1,0,0)	(1,1,0,1)	(1,1,1,0)	(1,1,1,1)

PRP-CCA

Definition (PRP-CCA)

Let k, n be positive integers, and let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a family of permutations. Let A be an adversary with access to two oracles. We define the following subroutines, experiment, and advantage function.

Subroutines

Subroutine *Initialize*

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

If $b = 1$ then $K \xleftarrow{\$} \{0,1\}^k$ else $p \xleftarrow{\$} \mathrm{Perm}(n)$

Subroutine $g(x)$

If $b = 1$ then return $E_K(x)$ else return $p(x)$

Subroutine $g^{-1}(x)$

If $b = 1$ then return $E_K^{-1}(x)$ else return $p^{-1}(x)$

Subroutine *Finalize*(d)

Return (d = b)

Experiment

Experiment $\mathbf{Exp}_E^{\mathrm{prp\text{-}cca}}(A)$

Initialize

$d \xleftarrow{\$} A^{g, g^{-1}}$

Return *Finalize*(d)

We define the **prp-cca advantage** of an adversary A mounting a chosen-ciphertext attack against E as

$$\mathbf{Adv}_E^{\mathrm{prp\text{-}cca}}(A) = 2 \cdot \Pr\left[\mathbf{Exp}_E^{\mathrm{prp\text{-}cca}}(A) \Rightarrow \mathsf{true}\right] - 1 .$$

A related security notion: PRF (Pseudorandom Function family)

Definition (PRF)

Let k, n be positive integers, and let $E : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a family of permutations. Let A be an adversary with access to an oracle. We define the following subroutines, experiment, and advantage function.

Subroutines

Subroutine *Initialize*
$b \xleftarrow{\$} \{0,1\}$
If $b = 1$ then $K \xleftarrow{\$} \{0,1\}^k$ else $f \xleftarrow{\$} \text{Func}(m, n)$

Subroutine $g(x)$
If $b = 1$ then return $F_K(x)$ else return $f(x)$

Subroutine $\mathit{Finalize}(d)$
Return (d = b)

Experiment

Experiment $\mathbf{Exp}_F^{\mathrm{prf}}(A)$

Initialize
$d \xleftarrow{\$} A^g$
Return $\mathit{Finalize}(d)$

We define the **prf advantage** of an adversary A attacking F as

$$\mathbf{Adv}_F^{\mathrm{prf}}(A) = 2 \cdot \Pr\left[\mathbf{Exp}_F^{\mathrm{prf}}(A) \Rightarrow \text{true}\right] - 1 \;.$$

Try using these definitions to analyze candidates

- Consider a family of permutations $F : \{0,1\}^{56} \times \{0,1\}^{64} \to \{0,1\}^{64}$ defined as follows: for any $K \in \{0,1\}^{56}$ and any $x \in \{0,1\}^{64}$,

$$F_K(x) = x \ .$$

- Consider a family of permutations $F : \{0,1\}^{64} \times \{0,1\}^{64} \to \{0,1\}^{64}$ defined as follows: for any $K \in \{0,1\}^{64}$ and any $x \in \{0,1\}^{64}$,

$$F_K(x) = K \oplus x \ .$$