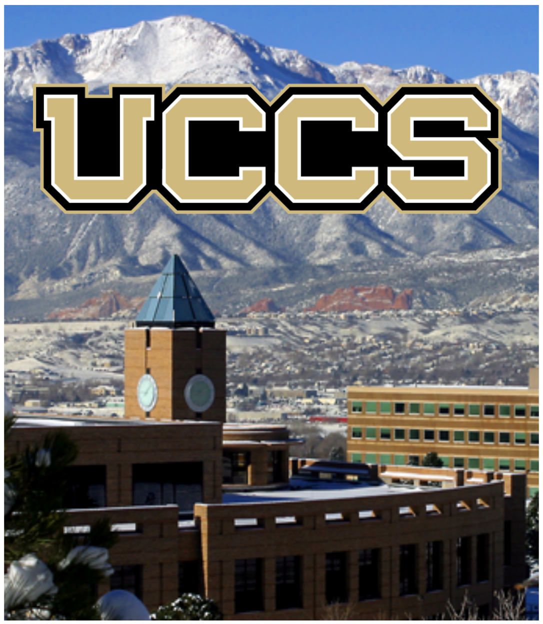


Symmetric Cryptography

3-DES and AES

Sang-Yoon Chang, Ph.D.

Module: 3-DES and AES

Double-DES

Meet-in-the-Middle Attack

Triple Data Encryption Standard (3-DES)

Advanced Encryption Standard (AES)

DES Security

Brute Force attacks in practice

Cryptanalytic attacks that can further reduce the complexity

Timing attacks on computation

Double-DES

DES Encryption (Enc) and Decryption (Dec)

Double-DES has two encryption stages
and two different keys (K_1, K_2):
$C = Enc(K_2, Enc(K_1, P));$
$P = Dec(K_1, Dec(K_2, C));$

Double-DES

DES Encryption (Enc) and Decryption (Dec)

Double-DES has two encryption stages
and two different keys (K_1, K_2):
$C = Enc(K_2, Enc(K_1, P));$
$P = Dec(K_1, Dec(K_2, C));$

Two keys (112 bits) ➔ 112 bits of entropy?

Meet in the Middle Attack

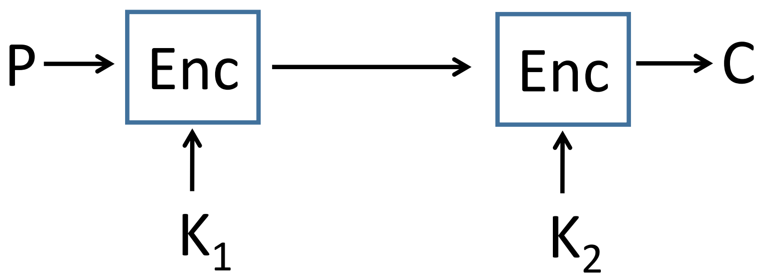
Applies for any block encryption cipher

$C = Enc(K_2, Enc(K_1, P))$

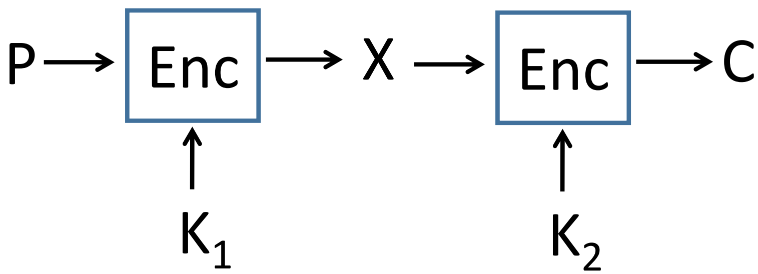
➔ $X = Enc(K_1, P) = Dec(K_2, C)$

Known plaintext attack

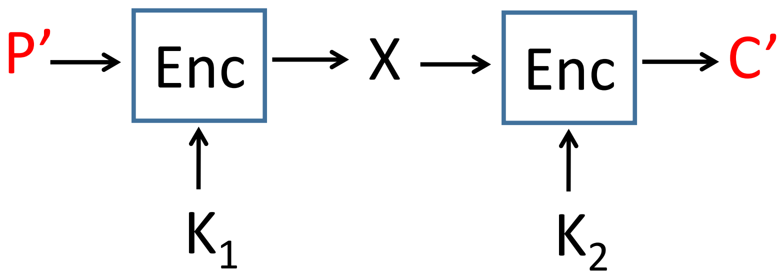
Meet in the Middle Attack



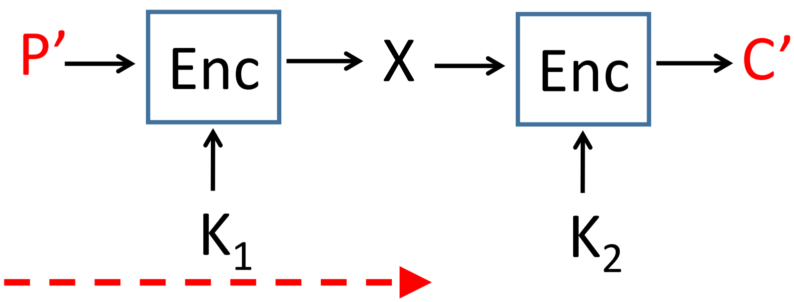
Meet in the Middle Attack



Meet in the Middle Attack



Meet in the Middle Attack



Compute and store 2^{56} P’→X mappings using different K_1’s

Meet in the Middle Attack

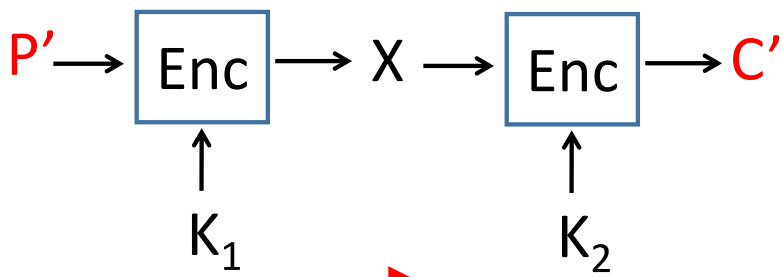
$$P' \rightarrow \boxed{\text{Enc}} \rightarrow X \rightarrow \boxed{\text{Enc}} \rightarrow C'$$

$$\text{Enc} \uparrow K_1 \qquad \text{Enc} \uparrow K_2$$

Compute and store 2^{56} $P' \rightarrow X$ mappings using different K_1's

Compute 2^{56} $C' \rightarrow X$ decryptions using K_2's

Meet in the Middle Attack



Compute and store 2^{56} P’→X mappings using different K_1’s

Compute 2^{56} C’→X decryptions using K_2’s

Compare X’s from two directions;
If the same, try with different known plaintexts (P’’,C’’)

Meet in the Middle Attack 

Attacker effort is $O(2^{56})$ and not $O(2^{112})$,

c.f., DES is $O(2^{55})$

Triple-DES

Triple-DES has three encryption stages:

$C = Enc(K_3, Dec(K_2, Enc(K_1, P)))$

$P = Dec(K_1, Enc(K_2, Dec(K_3, C)))$

Triple-DES

Triple-DES has three encryption stages:

$C = \mathrm{Enc}(K_3, \mathrm{Dec}(K_2, \mathrm{Enc}(K_1, P)))$

$P = \mathrm{Dec}(K_1, \mathrm{Enc}(K_2, \mathrm{Dec}(K_3, C)))$

Supports compatibility with single-DES

(Not recommended)

Triple-DES Keys

Key option 1: K_1, K_2, K_3 are independent

Key option 2: K_1, K_2 independent; $K_3 = K_1$

Key option 3: $K_3 = K_2 = K_1$
Equivalent to single-DES (ill-advised)

Triple-DES Keys

Key option 1: K_1, K_2, K_3 are independent

Key option 2: K_1, K_2 independent; $K_3 = K_1$

Key option 3: $K_3 = K_2 = K_1$

Equivalent to single-DES (ill-advised)

Triple-DES Keys

Key option 1: K_1, K_2, K_3 are independent

Key option 2: K_1, K_2 independent; $K_3 = K_1$

Makes the meet-in-the-middle attack effort $O(2^{112})$, c.f., double-DES $O(2^{56})$

Triple-DES Keys

Key option 1: K_1, K_2, K_3 are independent

Key option 2: K_1, K_2 independent; $K_3 = K_1$

➔ $C = Enc(K_1, Dec(K_2, Enc(K_1, P)))$

Makes the meet-in-the-middle attack effort $O(2^{112})$, c.f., double-DES $O(2^{56})$

Advanced Encryption Standard (AES)

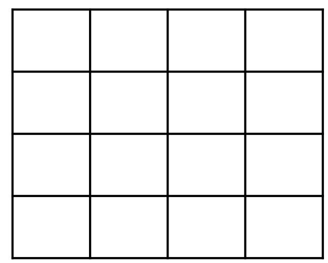
In 1997, US NIST call for ciphers
In 2001, standardized (FIPS PUB 197)

Replace DES and resist known attacks
Design simplicity
Speed and code compactness in CPU

Advanced Encryption Standard (AES)

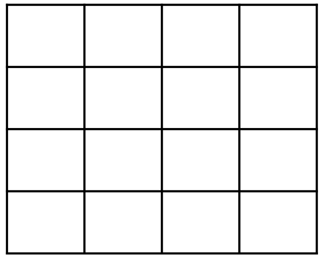
Byte-based processing and operations

128-bit (16B) block size with
128/192/256 bit key size



Not based on Feistel Cipher but based
on substitution and transposition

AES Rounds

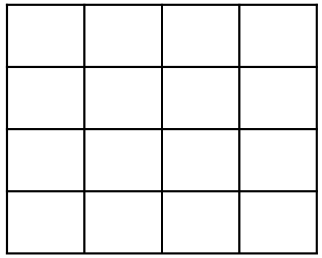


> ←Processes the data as 4x4 matrix of 16 bytes total (Each element is a Byte) "State array"

Iterated block cipher with rounds
(different round keys)

In addition to the initial round (XOR),
10 rounds for 128-bit key
12 rounds for 192-bit key
14 rounds for 256-bit key

AES Rounds

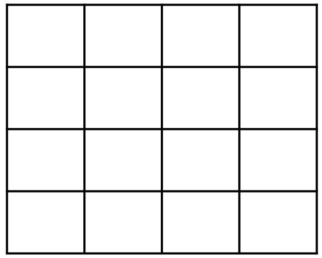


← Processes the data as 4x4 matrix of 16 bytes total (Each element is a Byte) "State array"

Except for initial (AddRoundKey only)
and final round (excluding MixColumns),
all rounds go through the following steps:

- SubBytes: Substitution using look-up table
- ShiftRows: Row-based transposition
- MixColumns: Column-based mapping
- AddRoundKey: XOR w/ 16B round key
  (KeyExapnsion: Round key generated)

AES



←Processes the data as 4x4 matrix of 16 bytes total (Each element is a Byte) "State array"

Only AddRoundKey uses key
(the cipher starts and ends with the step)

Additional AddRoundKey at the start,
and the final round is different

Each step is reversible

AES Decryption

Uses the round key in the reverse order

Reverse the steps order one-by-one

Except for AddRoundKey (XOR), the inverse functions are different for different steps
(Different decryption and encryption)