SYMMETRIC-KEY ENCRYPTION

Outline

1. Basic Concepts
2. The One-Time Pad
3. Stream Ciphers
4. The RC4 Stream Cipher
5. Block Ciphers
6. The Data Encryption Standard (DES)
7. Multiple Encryption
8. Modes of Operation
9. The Advanced Encryption Standard (AES)

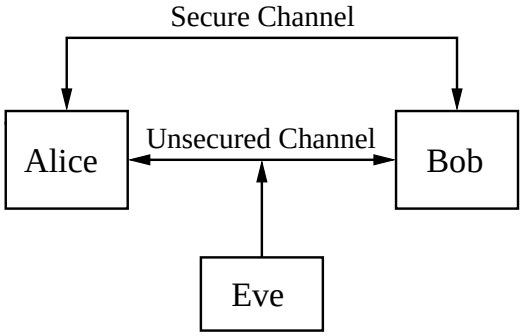
Basic Concepts

Definition: A *symmetric-key encryption scheme (SKES)* consists of:

- M – the plaintext space,
- C – the ciphertext space,
- K – the key space,
- a family of encryption functions, $E_k : M \to C$, $\forall k \in K$,
- a family of decryption functions, $D_k : C \to M$, $\forall k \in K$,

such that $D_k(E_k(m)) = m$ for all $m \in M$, $k \in K$.

Using a SKES to Achieve Confidentiality



1. Alice and Bob agree on a *secret key* $k \in K$ by communicating over the secure channel.
2. Alice computes $c = E_k(m)$ and sends the ciphertext c to Bob over the unsecured channel.
3. Bob retrieves the plaintext by computing $m = D_k(c)$.

The Simple Substitution Cipher

- M = all English messages.
- C = all encrypted messages.
- K = all permutations of the English alphabet.
- $E_k(m)$: Apply permutation k to m, one letter at a time.
- $D_k(c)$: Apply inverse permutation k^{-1} to c, one letter at a time.

Example:

k = a b c d e f g h i j k l m n o p q r s t u v w x y z
D N S E X K O J T A F P Y I Q U B R Z G V C H M W L

$m =$ the big dog, $c = E_k($the big dog$) =$ GJX NTO EQO.

Question: Is the simple substitution cipher a secure SKES?

What Does it Mean for a SKES to be Secure?

- What is the adversary's goal?
- What are the computational powers of the adversary?
- How does the adversary interact with the communicating parties?
- *Basic assumption*: The adversary knows everything about the SKES, except the particular key k chosen by Alice and Bob. (Avoid security by obscurity!!)
- *Security model*: Defines the computational abilities of the adversary, and how she interacts with the communicating parties.

Adversary's Interaction

- Passive attacks:
  - *Ciphertext-only attack*.
  - *Known-plaintext attack*: The adversary also knows some plaintext and the corresponding ciphertext.
- Active attacks:
  - *Chosen-plaintext attack*: The adversary can also choose some plaintext and obtains the corresponding ciphertext.
- Other attacks (not considered in this course):
  - *Side-channel attacks*: monitor the encryption and decryption equipment (timing attacks, power analysis attacks, electromagnetic-radiation analysis, etc.)
  - *Clandestine attacks*: bribery, blackmail, etc.

Computational Power of the Adversary

- *Information-theoretic security*: Eve has infinite computational resources.
- *Complexity-theoretic security*: Eve is a 'polynomial-time Turing machine'.
- *Computational security*: Eve has 18 Crays, 4000 Pentium PCs, and 200 DEC Alphas at her disposal. (Eve is "computationally bounded")

Adversary's Goal

1. Recover the secret key.
2. Systematically recover plaintext from ciphertext (without necessarily learning the secret key).
3. Learn *some* partial information about the plaintext from the ciphertext (other than its length).

- If the adversary can achieve 1 or 2, the SKES is said to be *totally insecure* (or *totally broken*).
- If the adversary cannot learn any partial information about the plaintext from the ciphertext (except possibly its length), the SKES is said to be *semantically secure*.

Definition of a Secure SKES

Definition: A symmetric-key encryption scheme is said to be *secure* if it is semantically secure against chosen-plaintext attack by a computationally bounded adversary.

To *break* a symmetric-key encryption scheme, the adversary has to accomplish the following:

1. Let c be a challenge ciphertext (generated by Alice or Bob using their secret key k).
2. During its computation, the adversary can select plaintext and obtains (from Alice or Bob) the corresponding ciphertext.
3. After a feasible amount of computation, the adversary obtains some information about the plaintext corresponding to c (other than the length of m).

Desirable Properties of a SKES

1. Efficient algorithms should be known for computing E_k and D_k (i.e. for encryption and decryption).
2. The key should be small (but large enough to render exhaustive key search infeasible).
3. The scheme should be secure.
4. The scheme should be secure even against the designer of the system.

Security of the Simple Substitution Cipher

Totally insecure against a chosen-plaintext attack. [Why?]

What about security under a ciphertext-only attack?

Is *exhaustive key search* possible?

- Given sufficient amounts of ciphertext c, decrypt c using all possible keys until c decrypts to a plaintext message which “makes sense”.
- In principle, 30 characters of ciphertext are sufficient on average to yield a unique plaintext that is a sensible English message. In practice, a few hundred characters are needed.

Security of the Simple Substitution Cipher (2)

Exhaustive search:

- Number of keys to try is $26! \approx 4 \times 10^{26} \approx 2^{88}$.
- If the adversary uses 10^6 computers, each capable of trying 10^9 keys per second, then exhaustive key search takes about 10^4 years.
- So, exhaustive key search is infeasible.

Of course, simple frequency analysis of ciphertext letters can be used to recover the key. So, the simple substitution cipher is totally insecure even against a ciphertext-only attack.

Work Factor

In this course:

- 2^{40} operations is considered *easy*.
- 2^{56} operations is considered *feasible*.
- 2^{64} operations is considered *barely feasible*.
- 2^{80} operations is considered *infeasible*.
- 2^{128} operations is considered *totally infeasible*.

Polyalphabetic Ciphers

Basic idea: Use several permutations, so a plaintext letter is encrypted to one of several possible ciphertext letters.

Example: *Vigenère cipher*:

- Key is an English word having no repeated letters
  e.g. $k =$ CRYPTO.
- Example of encryption:

$m =$	t	h	i	s		i	s		a		m	e	s	s	a	g	e
$+\, k =$	C	R	Y	P		T	O		C		R	Y	P	T	O	C	R
$c =$	V	Y	G	H		B	G		C		D	C	H	L	O	I	V

- Here, A=0, ..., Z=25; addition of letters is mod 26.
- Decryption is subtraction modulo 26.
- Frequency distribution of ciphertext letters is flatter (than for a simple substitution cipher).

Security of the Vigenère Cipher

- The Vigenère cipher is totally insecure against a chosen-plaintext attack. [Why?]
- What about security under a ciphertext-only attack?
- The key length l can be found as follows:
  - Make a guess for l and check your guess as follows.
    - ◊ Divide the ciphertext letters into l groups, $G_0, G_1, \ldots, G_{l-1}$, where the ith ciphertext letter is placed in group $G_i \bmod l$.
    - ◊ Examine the frequency distributions of letters in each group. If each distribution looks like the expected distribution of letters from sensible English text, then the key length guess is probably correct.

Security of the Vigenère Cipher (2)

- Once the key length l is known, then:
  - Notice that the ciphertext letters in each group G_i were obtained by applying a permutation that is a cyclic shift of the alphabet to the corresponding plaintext letters.
  - Use the frequency counts of ciphertext letters in G_i to make guesses for the ith letter of the key word.

The One-Time Pad

- Invented by Vernam in 1917 for the telegraph system
- Key is a *random* string of letters
- Example of encryption:

$m =$	t	h	i	s	i	s	a	m	e	s	s	a	g	e
$+\ k =$	Z	F	K	W	O	G	P	S	M	F	J	D	L	G
$c =$	S	M	S	P	W	Y	P	F	Q	X	C	D	R	K

- Note: The key is as long as the plaintext.
- The key should not be re-used:
  - If $c_1 = m_1 + k$ and $c_2 = m_2 + k$, then $c_1 - c_2 = m_1 - m_2$.
  - $c_1 - c_2$ depends only on the plaintext (and not on the key) and hence can leak information about the plaintext.
  - If m_1 is known, then m_2 can be easily computed.

Security of the One-Time Pad

- *Perfect secrecy*: The one-time pad is semantically secure against ciphertext-only attack by an adversary with infinite computational resources.
- This can be proven formally using concepts from information theory [Shannon 1949].
- The bad news: Shannon (1949) proved that if plaintexts are m-bit strings, then any symmetric-key encryption scheme with perfect secrecy must have $|K| \geq 2^m$.
- So, perfect secrecy (and the one-time pad) is fairly useless in practice.
- All is not lost: *stream ciphers*.

Stream Ciphers

- Basic idea: Instead of using a random key, use a "pseudorandom" key.
- A *pseudorandom bit generator* (PRBG) is a deterministic algorithm that takes as input a (random) *seed*, and outputs a longer "pseudorandom" sequence called the *keystream*.
- Convention: From now on, unless otherwise stated, messages and keys will be assumed to be bit strings.

Notation: $\oplus$ is bitwise exclusive-or (i.e., bitwise addition modulo 2). For example:

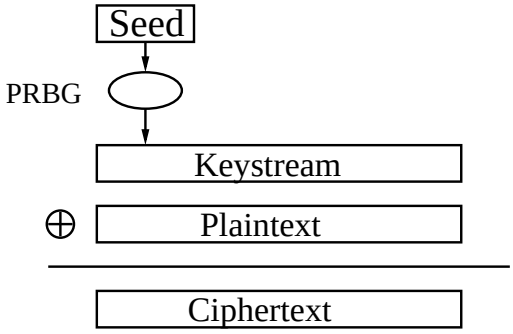
1011001010 $\oplus$ 1001001001 = 0010000011

Note that $x \oplus x = 0$ and $x \oplus y = y \oplus x$.

Hence if $x = y \oplus z$, then $x \oplus y = z$.

Stream Ciphers (2)

- Using a PRBG for encryption (a stream cipher):
  - The seed is the secret key shared by Alice and Bob.



- No more perfect secrecy – security depends on the quality of the PRBG.

Security Requirements for the PRBG

- The keystream should be "indistinguishable" from a random sequence (the *indistinguishability requirement*).
- If an adversary knows a portion c_1 of ciphertext and the corresponding plaintext m_1, then she can easily find the corresponding portion $k_1 = c_1 \oplus m_1$ of the keystream. Thus, given portions of the keystream, it should be infeasible to learn any information about the rest of the keystream (the *unpredictability requirement*).
- Aside: Don't use UNIX random number generators (rand and srand) for cryptography!
  ($X_0 =$ seed, $X_{i+1} = aX_i + b \bmod n$, $i \geq 0$.)
- We will study PRBGs in more detail later in the course....

The RC4 Stream Cipher

- (Alleged) proprietary algorithm (allegedly) designed by Ron Rivest of RSA Security (allegedly) in 1987.
- Reverse engineered in 1994.
- Widely used in commercial products: SSL (Netscape, IE), Lotus Notes, Windows password encryption, Adobe Acrobat, Oracle secure SQL, etc.
- Pros: Extremely simple; extremely fast; variable key sizes. No significant weakness has been found.
- Cons: Design criteria are proprietary (voodoo magic); not much public scrutiny until the year 2001.
- RC4 has two components: A *key scheduling algorithm*, and a *keystream generator*.

RC4 Key Scheduling Algorithm

In the following, $K[i]$, $\overline{K}[i]$ and $S[i]$ are 8-bit integers.

Input: Secret key $K[0], K[1], \ldots, K[d-1]$. (Keysize is $8d$ bits.)
Output: 256-long array: $S[0], S[1], \ldots, S[255]$.

For i from 0 to 255 do:
    $S[i] \leftarrow i$
    $\overline{K}[i] \leftarrow K[i \bmod d]$
$j \leftarrow 0$
For i from 0 to 255 do:
    $j \leftarrow (\overline{K}[i] + S[i] + j) \bmod 256$
    Swap($S[i], S[j]$)

[S is a "random" permutation of $\{0, 1, 2, \ldots, 255\}$ that is generated from the secret key]

RC4 Keystream Generator

Input: 256-long byte array: $S[0], S[1], \ldots, S[255]$.
Output: Keystream.

$i \leftarrow 0$; $j \leftarrow 0$
While keystream bytes are required do:
$i \leftarrow (i + 1) \bmod 256$
$j \leftarrow (S[i] + j) \bmod 256$
Swap($S[i], S[j]$)
$t \leftarrow (S[i] + S[j]) \bmod 256$
Output($S[t]$)

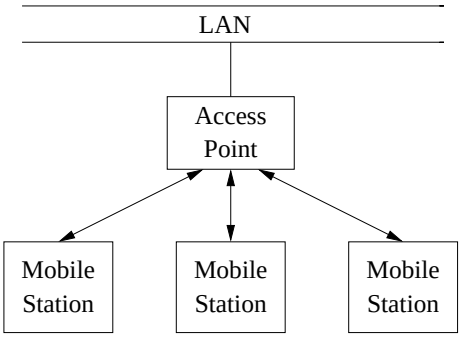
[The keystream bytes are xored with the plaintext bytes]

Wireless Security

- Wireless networks becoming prevalent.
- Popular standards for wireless networks:
  - IEEE 802.11 (longer range, higher speeds, commonly used for wireless LANs).
  - Bluetooth (short range, low speed).
- New security concerns:
  - More attack opportunities (no need for physical access).
  - Attack from a distance ($>$ 1 km with good antennae).
  - No physical evidence of attack.

Case Study: IEEE 802.11 Security

- IEEE 802.11 standard for wireless LAN communications includes a protocol called *Wired Equivalent Privacy* (WEP). IEEE 802.11 and WEP are in widespread use today.
- WEP's goal is (only) to protect link-level data during wireless transmission between mobile stations and access points.



Main Security Goals of WEP

1. *Confidentiality*: Prevent casual eavesdropping.
   - RC4 is used for encryption.
2. *Data Integrity*: Prevent tampering with transmitted messages.
   - An 'integrity checksum' is used.
3. *Access Control*: Protect access to a wireless network infrastructure.
   - Discard all packets that are not properly encrypted using WEP.

Description of WEP Protocol

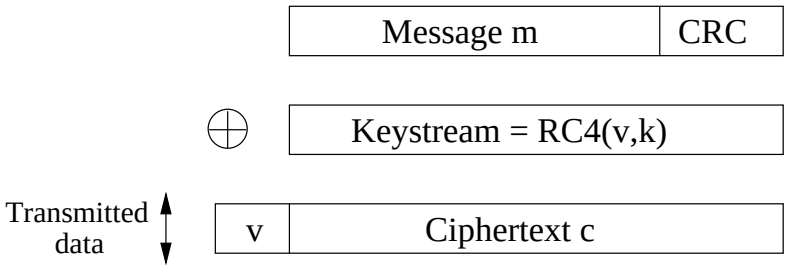
- Mobile station shares a secret key k with access point.
  - k is either 40 bits or 104 bits in length.
  - The standard does not specify how the key is to be distributed.
  - In practice, one shared key per LAN is common; this key is manually injected into each access point and mobile station; the key is not changed very frequently.
- Messages are divided into *packets* of some fixed length (e.g. 1500 bytes).
- WEP uses a per-packet 24-bit IV v to process each packet. WEP does not specify how the IVs are managed. In practice:
  - A random IV is generated for each packet; or
  - The IV is set to 0 and incremented by 1 for each use.

Description of WEP Protocol (2)

To send a packet m, an entity does the following:

1. Select a 24-bit IV v.
2. Compute a 32-bit *checksum*: $S = \mathsf{CRC}(m)$.
   - 802.11 specifies that a CRC-32 checksum be used. CRC-32 is *linear*. That is, for any two messages m_1 and m_2 of the same bitlength,
     $$\mathsf{CRC}(m_1 \oplus m_2) = \mathsf{CRC}(m_1) \oplus \mathsf{CRC}(m_2).$$
     (The details of CRC-32 are not important to us)
3. Compute $c = (m \| S) \oplus \mathsf{RC4}(v \,\|\, k)$.
   - $\|$ denotes concatenation.
   - $(v \,\|\, k)$ is the key used in the RC4 stream cipher.
4. Send (v, c) over the wireless channel.

Description of WEP Protocol (3)



The receiver of (v, c) does the following:

1. Compute $(m \,\|\, S) = c \oplus \mathsf{RC4}(v \,\|\, k)$.
2. Compute $S' = \mathsf{CRC}(m)$; reject the packet if $S' \neq S$.

Question: Are confidentiality, data integrity, access control achieved?
Answer: NO! [Borisov, Goldberg & Wagner; 2001]

Problem: IV Collision

- Suppose that two packets (v, c) and (v, c') use the same IV. Let m, m' be the corresponding plaintexts. Then $c \oplus c' = m \oplus m'$.
- If m is known, then m' is immediately available.
- If m is not known, then one may be able to use the expected distribution of m and m' to discover information about them. (Much of network traffic contents is predictable.)

Finding IV Collisions

- Since there are only 2^{24} choices for the IV, collisions are guaranteed after enough time — a few days on a busy network (5 Mbps).
- If IVs are randomly selected, then one can expect a collision after about 2^{12} packets. [The birthday paradox]
- Collisions are more likely if keys k are long-lived and the same key is used for multiple mobile stations in a network.
- Thus, WEP does not provide a high degree of confidentiality.

Problem: Checksum is Linear

- CRC-32 is used to check integrity. This is fine for random errors, but not for deliberate ones.
- It is easy to make controlled changes to (encrypted) packets:
  - Suppose (v, c) is an encrypted packet.
  - Let $c = \text{RC4}(v\|k) \oplus (m\|S)$, where k, m, S are unknown.
  - Let $m' = m \oplus \Delta$, where Δ is a bit string. (The 1's in Δ correspond to the bits of m an attacker wishes to change.)
  - Let $c' = c \oplus (\Delta\|\text{CRC}(\Delta))$.
  - Then (v, c') is a valid encrypted packet for m'. [Exercise: Prove this]
- Thus, WEP does not provide data integrity.

Problem: Integrity Function is Unkeyed

- Suppose that an attacker learns the plaintext m corresponding to a single encrypted packet (v, c).
- Then, the attacker can compute the RC4 keystream $\text{RC4}(v\|k) = c \oplus (m\|\text{CRC}(m))$.
- Henceforth, the attacker can compute a valid encrypted packet for *any* plaintext m' of her choice: (v, c'), where $c' = \text{RC4}(v\|k) \oplus (m'\|\text{CRC}(m'))$.
- Thus, WEP does not provide access control.

Required Reading: Sections 1, 2, 3, 4.1, 4.2, 6 of "Intercepting mobile communications: The insecurity of 802.11", by N. Borisov, I. Goldberg and D. Wagner.

A More Devastating Attack

- Fluhrer, Mantin & Shamir, 2001.
- Assumptions:
  1. The same 104-bit key k is used for a long period of time. [Most products do this.]
  2. The IV is incremented for each packet, or a random IV is selected for each packet. [Most products do this.]
  3. The first plaintext byte of each packet (i.e. the first byte of each m) is known to the attacker. [Most wireless protocols prepend the plaintext with some header bytes which are non-secret.]
- *Attack*: A *passive* adversary who can collect about 5,000,000 encrypted packets can very easily recover k (and thus totally break the system). [Details not covered in this course.]

Implementing the Fluhrer/Mantin/Shamir Attack

- The attack can be easily mounted in practice:
  - Can buy a $100 wireless card and hack drivers to capture (encrypted) packets.
  - On a busy wireless network (5Mbps), 5 million packets can be captured in a few hours, and then k can be immediately computed.
- Implementation details: A. Stubbefield, J. Ionnidis, A. Rubin, “Using the Fluhrer, Mantin and Shamir attack to break WEP”, AT&T Technical Report, August 2001.
- Script kiddies:
  - *Aircrack-ng*: `//www.aircrack-ng.org/doku.php`
  - *WEPCrack*: `sourceforge.net/projects/wepcrack`
- *aircrack-ptw*: Breaks WEP in under 60 seconds (only about 40,000 packets are needed).

Lessons Learned

1. *Bacon ice-cream is bad*: RC4 is a very good (secure) stream cipher. However, it was improperly used in WEP and the result was a highly insecure communications protocol. Note that RC4 as used in SSL is not vulnerable to the attacks we described (more on this later...)
2. *Designing security is hard*:
   - Designing cryptographic protocols is complicated and difficult.
   - Clearly state your security objectives.
   - Hire experts to design your security.
   - Make your protocols available for public scrutiny.

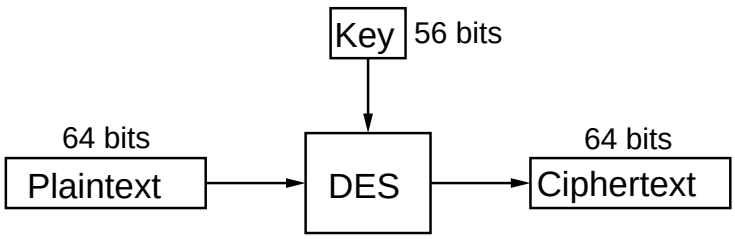
IEEE 802.11 Update

`http://en.wikipedia.org/wiki/Wi-Fi_Protected_Access`

- Wi-fi Protected Access (*WPA*) [temporary replacement]
  - Uses a master key, from which 128-bit session keys are periodically generated.
  - The 128-bit session key is combined with a 48-bit IV to form the RC4 key.
  - A Message Authentication Code (MAC) algorithm is used instead of CRC.
  - WPA is compatible with many (but not all) WEP access points.
- *WPA2*
  - Implements the new *IEEE 802.11i* standard (2004).
  - Uses the AES block cipher instead of RC4.
  - Supported in Microsoft Windows XP as of May 1 2005.

Block Ciphers

- A *block cipher* is a SKES which breaks up the plaintext into blocks of a fixed length (e.g. 128 bits), and encrypts the blocks one at a time.
- In contrast, a *stream cipher* encrypts the plaintext one character (usually a bit) at a time.
- Canonical example of a block cipher:
  The *Data Encryption Standard (DES)*



Key size: 56 bits; Size of key space: 2^{56}; Block size: 64 bits.

Brief History of Block Ciphers

- Late 1960's: Feistel network and LUCIFER designed at IBM.
- 1972: NBS (now *NIST*: National Institute of Standards and Technology) solicits proposals for encryption algorithms for the protection of computer data.
- 1974: IBM develops DES.
- 1975: *NSA* (National Security Agency) "fixes" DES
  - Reduces the key size from 128 bits to 56 bits.
- 1977: DES adopted as US Federal Information Processing Standard (FIPS 46).
- 1981: DES adopted as a US banking standard (ANSI X3.92).

Brief History of Block Ciphers (2)

- 1997: NIST begins the AES (Advanced Encryption Standard) competition.
- 1999: 5 finalists for AES announced.
- 2001: Rijndael adopted for AES (FIPS 197).
  - AES has three key sizes: 128, 192 and 256 bits.
- 2009:
  - No significant weaknesses found with AES (as yet).
  - AES uptake is rapidly increasing.
  - However, DES (and Triple-DES) is still widely deployed.

The National Security Agency (NSA)

- `www.nsa.gov`
- Founded in 1952.
- Budget: Unknown; Number of employees: Unknown.
- Signals Intelligence (SIGINT): produce foreign intelligence information.
- Information Systems Security (INFOSEC): protects all classified and sensitive information that is stored or sent through US government equipment.
- Very influential in setting US government export policy for cryptographic products (especially encryption).
- Canadian counterpart: Communications Security Establishment (CSE); `www.cse.dnd.ca`

Some Desirable Properties of Block Ciphers

- Security:
  - *Diffusion*: each ciphertext bit should depend on all plaintext bits.
  - *Confusion*: the relationship between key and ciphertext bits should be complicated.
  - *Key size*: should be small, but large enough to preclude exhaustive key search.
- Efficiency:
  - High encryption and decryption rate.
  - Simplicity (easier to implement and analyze).
  - Suitability for hardware or software.

Feistel Ciphers: A Class of Block Ciphers

Components of a Feistel cipher:

- Parameters: n (half the block length), h (number of rounds), l (key size).
- $M = \{0,1\}^{2n}$, $C = \{0,1\}^{2n}$, $K = \{0,1\}^{l}$.
- A *key scheduling algorithm* which determines *subkeys* $k_1, k_2, \ldots, k_h$ from a key k.
- Each subkey k_i defines a *component function* $f_i : \{0,1\}^n \to \{0,1\}^n$.

Components of a Feistel Cipher (cont'd)

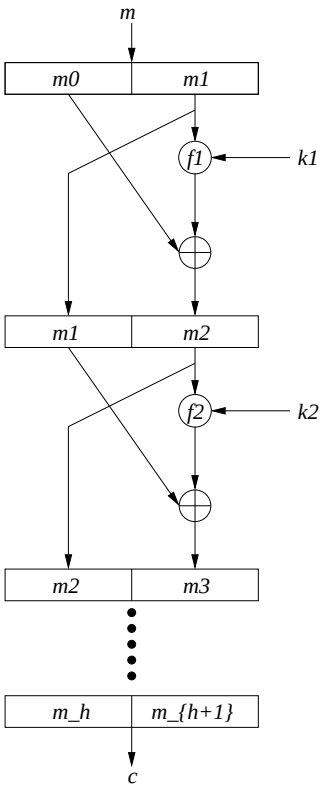
Encryption takes h rounds:

- Plaintext is $m = (m_0, m_1)$, where $m_i \in \{0,1\}^n$.
- Round 1: $(m_0, m_1) \to (m_1, m_2)$, where $m_2 = m_0 \oplus f_1(m_1)$.
- Round 2: $(m_1, m_2) \to (m_2, m_3)$, where $m_3 = m_1 \oplus f_2(m_2)$.
- .....
- Round h: $(m_{h-1}, m_h) \to (m_h, m_{h+1})$, where $m_{h+1} = m_{h-1} \oplus f_h(m_h)$.
- Ciphertext is $c = (m_h, m_{h+1})$.

Decryption: Given $c = (m_h, m_{h+1})$ and k, find $m = (m_0, m_1)$:

- Compute $m_{h-1} = m_{h+1} \oplus f_h(m_h)$.
- Similarly, compute $m_{h-2}, \ldots, m_1, m_0$.

The Feistel Ladder



Feistel Cipher (notes)

- No restrictions need be placed on the functions f_i in order for the encryption procedure to be invertible.
- Underlying principle: Take something "simple" and use it several times; hope that the result is "complicated".
- Implementation:
  - Encryption: Only need to implement one round; the same code can be used for each round.
  - Decryption: Can use the code for encryption. (Use subkeys in reverse order.)
- Two examples of Feistel ciphers we will study:
  - New Data Seal.
  - DES.

The New Data Seal (NDS)

- Feistel cipher with $n = 64$, $h = 16$.
- Key is a random function $S_k : \{0,1\}^8 \to \{0,1\}^8$ ($l = 2048$; exhaustive key search is infeasible).

  Note: S_k can be represented by a table.

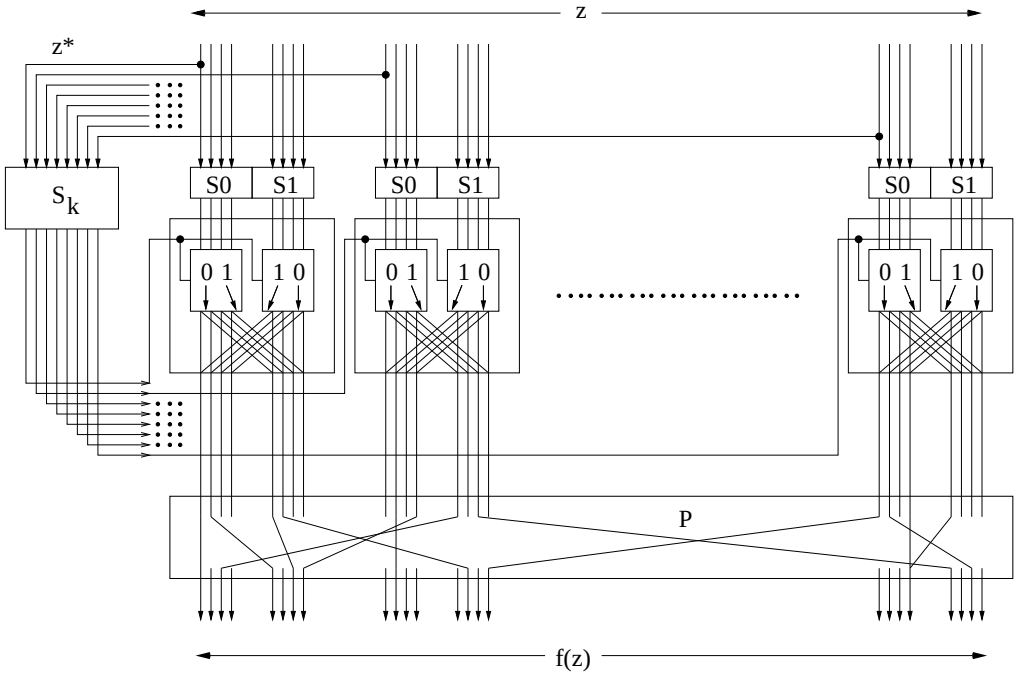
  Example:

r	0	1	2	3	4	5	$\cdots$	255
$S_k(r)$	73	106	5	7	73	119	$\cdots$	210

- Each subkey is S_k itself (so we will write f for f_i).
- The component function $f : \{0,1\}^{64} \to \{0,1\}^{64}$ is defined in the following two slides.

The NDS Component Function f

The NDS component function $f : \{0,1\}^{64} \to \{0,1\}^{64}$.



The NDS Component Function f

Input is $z \in \{0,1\}^{64}$. Output is $f(z) \in \{0,1\}^{64}$.

1. Divide z into 8 bytes: $z = (z^1, z^2, \ldots, z^8)$.
2. Divide each byte z^j into two nibbles: (n_1^j, n_2^j).
3. Apply $S_0 : \{0,1\}^4 \to \{0,1\}^4$ to n_1^j to get p_1^j $(1 \leq j \leq 8)$.
   Apply $S_1 : \{0,1\}^4 \to \{0,1\}^4$ to n_2^j to get p_2^j $(1 \leq j \leq 8)$.
   (S_0, S_1 are fixed and public knowledge)
4. Let z^* be the byte obtained by taking the first bit of each byte of z. Compute the byte $t = S_k(z^*)$.
5. If the jth bit of t is 1, then swap p_1^j and p_2^j $(1 \leq j \leq 8)$.
6. Permute (rearrange) the resulting 64 bits as determined by P. (P is fixed and public knowledge)

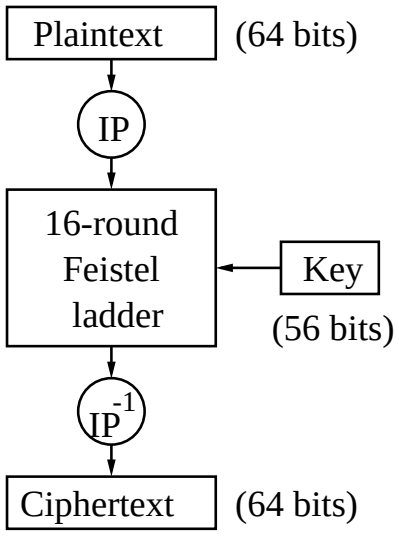
Chosen-Plaintext Attack on NDS

- An adversary can easily determine the secret key after obtaining the ciphertexts for about 32,000 carefully-chosen plaintexts.
- The attack shows that NDS is insecure.
- The attack also demonstrates the importance of using different subkeys in the rounds of a Fiestel cipher.
- For a description of the attack see your notes from class.

The Data Encryption Standard

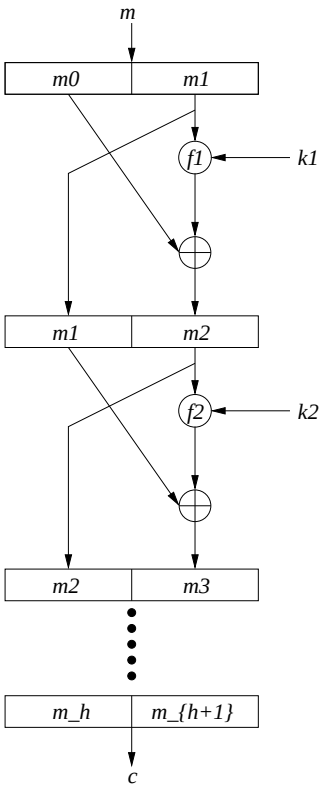
- Designed in 1973/74 by IBM.
- “Fixed” by NSA in 1975 (keysize reduced from 128 bits to 56 bits).
- Adopted in 1977 by NIST as a US government FIPS for encryption of unclassified data.
- Adopted in 1981 as a US banking standard.
- Still the most widely used block cipher.
- Design principles are still classified, making analysis difficult.
- Hundreds of research papers written on security of DES.

Overview of DES



(IP = Initial Permutation)

The Feistel Ladder

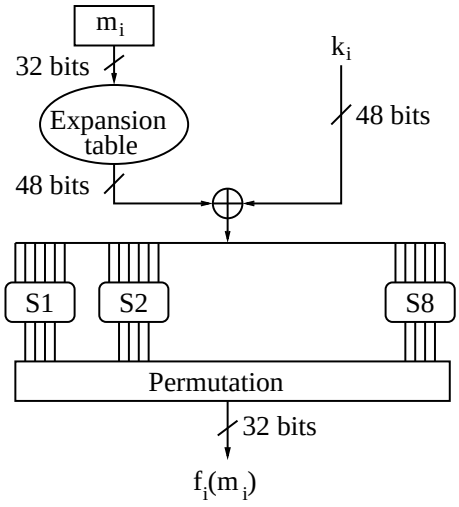


Description of DES

- Feistel cipher with $n = 32$, $h = 16$, $l = 56$.
- Key scheduling algorithm:
  - Selects 16 48-bit subkeys $k_1, \ldots, k_{16}$ from the key k.
  - Each subkey consists of a (fixed) selection of 48 bits of k.
- S-boxes (S1, S2, S3, S4, S5, S6, S7, S8):
  - The only part of DES that are non-linear.
  - The security of DES crucially depends on their choice.
  - DES with randomly selected S-boxes is easy to break.

Description of DES (2)

Component function $f_i : \{0,1\}^{32} \rightarrow \{0,1\}^{32}$:



Note: The IP, key scheduling algorithm, Expansion table, S-boxes, and Permutation are fixed and public knowledge.

Performance

[Wei Dai] `www.cryptopp.com/benchmarks.html`
Speed benchmarks for a software implementation on an Intel Core 2 1.83 GHz processor.

SKES	Block size (bits)	Key size (bits)	Speed
DES	64	56	34 Mbits/sec
3DES	64	112	13 Mbits/sec
AES	128	128	99 Mbits/sec
RC4	—	variable	128 Mbits/sec
SKIPJACK	80	80	10 Mbits/sec
DES	64	56	10+ Gbits/sec (ASIC chip)

DES Problem 1: Small Key Size

- Exhaustive search on key space takes 2^{56} steps and can be easily parallelized.
- DES challenges from RSA Security (3 known PT/CT pairs):

`The unkn`	`own mess`	`age is:`	`????????`

  - June 1997: Broken by Internet search (3 months).
  - July 1998: Broken in 3 days by DeepCrack machine (1800 chips; $250,000).
  - Jan 1999: Broken in 22 hrs, 15 min (DeepCrack + `distributed.net`).
- RC5 64-bit challenge:
  - July 2002: Broken in 1757 days.
  - Participation by 331,252 individuals.
- In progress (`distributed.net`): RC5 72-bit challenge.

DES Problem 2: Small Block Size

- *Birthday paradox*:
  - An urn has n balls numbered 1 to n.
  - Balls are drawn one at a time, with replacement.
  - The expected number of draws before a repeat (*collision*) is approximately $\sqrt{\pi n/2}$.
  - For $n = 365$, this number is approximately 24.
- If plaintext blocks are distributed "uniformly at random", then the expected number of ciphertext blocks observed before a collision occurs is $\approx 2^{32}$.
  - Hence the ciphertext reveals *some* information about the plaintext.
- Small block size is also damaging to some authentication applications (more on this later).

Sophisticated Attacks on DES

Differential cryptanalysis [Biham & Shamir 1989]:

- Recovers key given 2^{47} chosen plaintext/ciphertext pairs.
- DES was designed to resist this attack.
- Differential cryptanalysis has been more effective on some other block ciphers.

Linear cryptanalysis [Matsui 1993]:

- Recovers key given 2^{43} known plaintext/ciphertext pairs.
- Storing these pairs takes 131,000 Gbytes.
- Implemented in 1993: 10 days on 12 machines.

Implementation Attacks on DES

Power analysis attacks:

- Kocher, Jaffe & Jun 1999.
- Processor power consumption depends on instruction.
- Measure power consumption of instructions executed in 16th round of DES.
- $\approx$1000 encryptions suffice to expose the secret key.

Differential fault analysis (DFA) attacks:

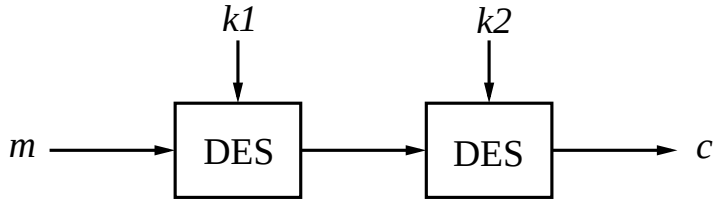
- Biham & Shamir 1997.
- Attack: induce random errors in 16th round of DES.
- $\approx$200 erroneous decryptions expose the secret key.

Multiple Encryption

- Recall that the only weaknesses known in DES are the obvious ones: small key size and small block size.
- Question: How can one construct a more secure block cipher from DES? (i.e. without changing the internals of DES.)
- *Multiple encryption*: Re-encrypt the ciphertext one or more times using independent keys. Hope that this increases the effective key length.
- Multiple encryption does not always result in increased security.
  Example: If E_π denotes the simple substitution cipher with key π, then is $E_{\pi_1} \circ E_{\pi_2}$ any more secure than E_π?

Double Encryption

- *Double-DES*. Key is $k = (k_1, k_2)$, $k_1, k_2 \in_R \{0,1\}^{56}$.
- *Encryption*: $c = E_{k_2}(E_{k_1}(m))$.
  (E = DES encryption, E^{-1} = DES decryption)



- *Decryption*: $m = E_{k_1}^{-1}(E_{k_2}^{-1}(c))$.
- Key size of Double-DES is $l = 112$, so exhaustive key search takes 2^{112} steps (infeasible).
- Note: Block length is unchanged.

Meet-In-The-Middle Attack on Double-DES

Main idea: If $c = E_{k_2}(E_{k_1}(m))$, then $E_{k_2}^{-1}(c) = E_{k_1}(m)$.

Input: 3 known PT/CT pairs (m_1, c_1), (m_2, c_2), (m_3, c_3).
Output: The secret key (k_1, k_2).

1. For each $h_2 \in \{0,1\}^{56}$:
   (a) Compute $E_{h_2}^{-1}(c_1)$, and store $[E_{h_2}^{-1}(c_1), h_2]$ in a table sorted by first component.
2. For each $h_1 \in \{0,1\}^{56}$ do the following:
   (a) Compute $E_{h_1}(m_1)$.
   (b) Search for $E_{h_1}(m_1)$ in the table. (We say that $E_{h_1}(m_1)$ *matches* table entry $[E_{h_2}^{-1}(c_1), h_2]$ if $E_{h_1}(m_1) = E_{h_2}^{-1}(c_1)$.)
   (c) For each match $[E_{h_2}^{-1}(c_1), h_2]$ in the table, check if $E_{h_2}(E_{h_1}(m_2)) = c_2$; if so then check if $E_{h_2}(E_{h_1}(m_3)) = c_3$. If both checks pass, then output (h_1, h_2) and STOP.

Number of PT/CT Pairs Needed

Number of known plaintext/ciphertext pairs needed for unique key determination:

Question: Let E be a block cipher with key space $K = \{0,1\}^l$, and plaintext and ciphertext space $\{0,1\}^L$.

Let $k' \in K$ be the secret key chosen by Alice and Bob, and let (m_i, c_i), $1 \leq i \leq t$, be known plaintext/ciphertext pairs, where the plaintext m_i are pairwise distinct.
(Note that $c_i = E_{k'}(m_i)$ for all $1 \leq i \leq t$)

Then how large should t be to ensure (with probability very close to 1) that there is only one key $k \in K$ such that $E_k(m_i) = c_i$ for all $1 \leq i \leq t$?

Number of PT/CT Pairs Needed (2)

- For each $k \in K$, the encryption function $E_k : \{0,1\}^L \to \{0,1\}^L$ is a permutation.
- We make the *heuristic assumption* that for each $k \in K$, E_k is a random function (i.e., a randomly selected function). This assumption is certainly false since E_k is *not* random, and because a random function is almost certainly now a permutation. Nonetheless, it turns out that the assumption is good enough for our analysis.
- Now, fix $k \in K$, $k \neq k'$. Then the probability that $E_k(m_i) = c_i$ for all $1 \leq i \leq t$ is
$$\underbrace{\frac{1}{2^L} \cdot \frac{1}{2^L} \cdots \frac{1}{2^L}}_{t} = \frac{1}{2^{Lt}}.$$
- Thus the expected number of $k \in K$ (not including k') for which $E_k(m_i) = c_i$ for all $1 \leq i \leq t$ is
$$FK = \frac{2^l - 1}{2^{Lt}}.$$

Meet-In-The-Middle Attack on Double DES

Let E be the DES encryption function, so Double-DES encryption is $c = E_{k_2}(E_{k_1}(m))$.

1. If $l = 112$, $L = 64$, $t = 3$, then $FK \approx 1/2^{80}$.
   Thus if a Double-DES key (h_1, h_2) is found for which $E_{h_2}(E_{h_1}(m_i)) = c_i$ for $i = 1, 2, 3$, then with very high probability we have $(h_1, h_2) = (k_1, k_2)$.
2. If $l = 112$, $L = 64$, $t = 1$, then $FK \approx 2^{48}$.
   Thus the expected number of Double-DES keys (h_1, h_2) for which $E_{h_2}(E_{h_1}(m_1)) = c_1$ is 2^{48}.
3. Of the 2^{48} keys (h_1, h_2) satisfying $E_{h_2}(E_{h_1}(m_1)) = c_1$, the expected number of keys which also satisfy $E_{h_2}(E_{h_1}(m_2)) = c_2$ is $\approx 2^{48}/2^{64} = 1/2^{16}$.

Analysis of the Meet-In-The-Middle Attack

Analysis:

- Number of DES operations is $\approx 2^{56} + 2^{56} + 2 \cdot 2^{48} \approx 2^{57}$.
  (We are not counting the time to do the sorting and searching)
- Space requirements: $2^{56}(64 + 56)$ bits $\approx 983,040$ Tbytes.

Time-memory tradeoff. [Exercise] The attack can be modified to decrease the storage requirements at the expense of time:

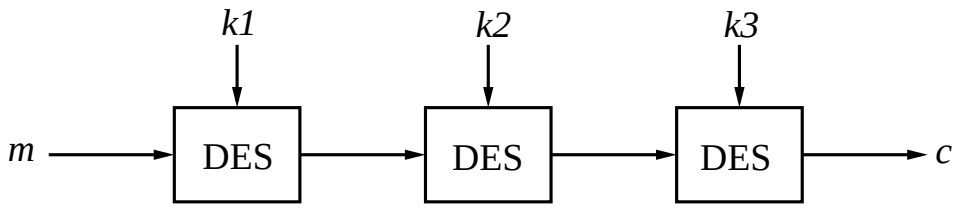
- Time: 2^{56+s} steps; memory: 2^{56-s} units, $1 \leq s \leq 55$.

Conclusions:

- Double-DES has the same effective key length as DES.
- Double-DES is not much more secure than DES.

Triple-Encryption

- *Triple-DES*. Key is $k = (k_1, k_2, k_3)$, $k_1, k_2, k_3 \in_R \{0,1\}^{56}$.
- *Encryption*: $c = E_{k_3}(E_{k_2}(E_{k_1}(m)))$.
  (E = DES encryption, E^{-1} = DES decryption)



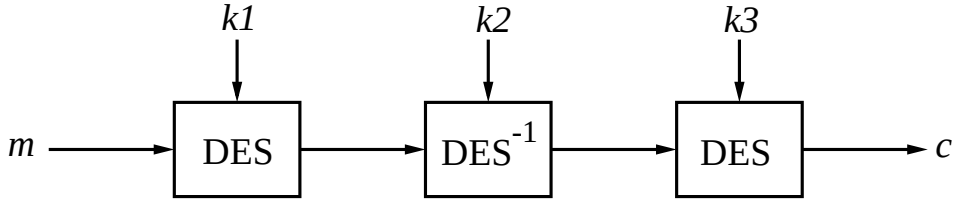
- *Decryption*: $m = E_{k_1}^{-1}(E_{k_2}^{-1}(E_{k_3}^{-1}(c)))$.
- Key length of Triple-DES is $l = 168$, so exhaustive key search takes 2^{168} steps (infeasible).

Triple-Encryption (2)

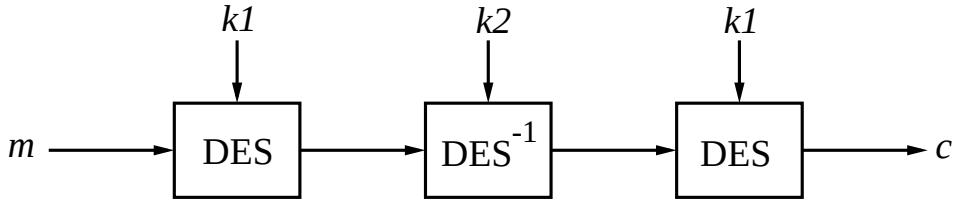
- Meet-in-the-middle attack takes $\approx 2^{112}$ steps. [Exercise]
- So, the effective key length of Triple-DES against exhaustive key search is $\leq$ 112 bits.
- No *proof* that Triple-DES is more secure than DES.
- Note: Block length is unchanged.
- Possibility of dictionary attacks.
  - Adversary stores a large table (of size $\leq 2^{64}$) of (m, c) pairs.
  - To prevent this attack: change secret keys periodically.
- Triple-DES is widely deployed.

Some Variants

EDE Triple-DES: for backward compatibility with DES



Two-key Triple-DES:



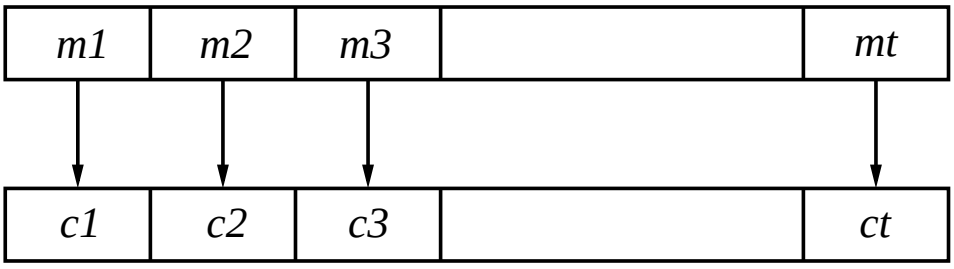
Effective key length is 56 bits (under a chosen-plaintext attack). [Exercise (hard)]

Block Cipher Modes of Operation

- Plaintext message is $m = m_1, m_2, \ldots, m_t$, where each m_i is an L-bit block.
- Question: How should we use a block cipher $E_k : \{0,1\}^L \rightarrow \{0,1\}^L$ to encrypt m?

Electronic Codebook (ECB) Mode

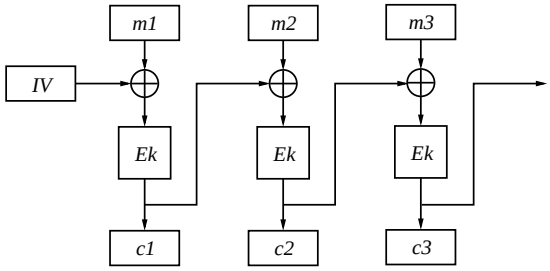
- *Encrypt* blocks independently, one at a time: $c = c_1, c_2, \ldots, c_t$, where $c_i = E_k(m_i)$.



- *Decryption*: $m_i = E_k^{-1}(c_i)$, $i = 1, 2, \ldots, t$.
- Drawback: Identical plaintexts result in identical ciphertexts (under the same key), and thus ECB encryption is not (semantically) secure against chosen plaintext attacks. [Why?]

Cipher Block Chaining (CBC) Mode

- *Encryption*: Select $c_0 \in_R \{0,1\}^L$ (c_0 is a random non-secret IV). Then compute $c_i = E_k(m_i \oplus c_{i-1})$, $i = 1, 2, \ldots, t$.



- Ciphertext: $c_0, c_1, c_2, \ldots, c_t$.
- *Decryption*: $m_i = E_k^{-1}(c_i) \oplus c_{i-1}$, $i = 1, 2, \ldots, t$.
- Identical plaintexts with different IVs result in different ciphertexts and for this reason CBC encryption is (semantically) secure against chosen-plaintext attacks (for a well chosen block cipher E).

The Advanced Encryption Standard (AES)

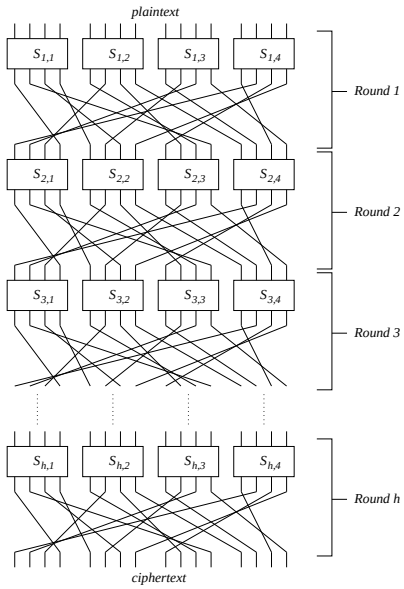
- `www.nist.gov/aes`
- September 1997: Call issued for AES candidate algorithms.
- Requirements:
  - Key sizes: 128, 192 and 256 bits.
  - Block size: 128 bits.
  - Efficient on both hardware and software platforms.
  - Availability on a worldwide, non-exclusive, royalty-free basis.

The AES Process

- August 1998: 15 submissions in Round 1.
- August 1999: 5 finalists selected by NIST:
  - MARS, RC6, Rijndael, Serpent, Twofish.
- 1999: NSA performed a hardware efficiency comparison.
- October 2 2000: *Rijndael* was selected.
- December 2001: The AES standard is officially adopted (FIPS 197).
- Rijndael is an iterated block cipher. It is not a Feistel cipher, but is an example of a substitution-permutation network.

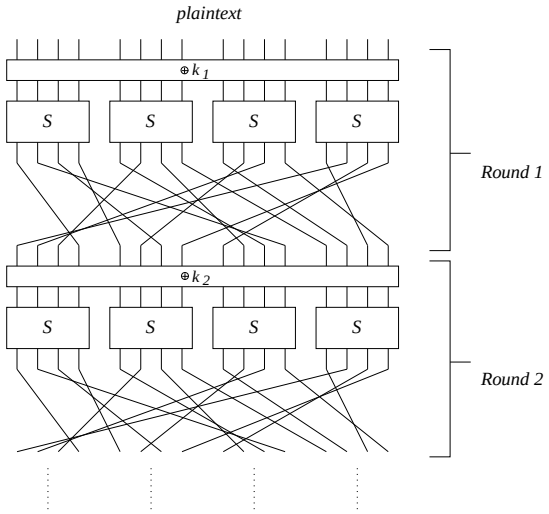
Substitution-Permutation Networks (1)

A *substitution-permutation network* (SPN) is an iterated block cipher where a round consists of a *substitution* operation followed by a *permutation* operation.



Substitution-Permutation Networks (2)

- The key k is used to influence the result of the substitution step.
- One way to do this is to XOR the S-box inputs with key bits before the S-box is applied.
- From k, derive round keys, $k_1, k_2, \ldots, k_h, k_{h+1}$.
  (h denotes the number of rounds)



Substitution-Permutation Networks (3)

- By XOR-ing an additional round key, k_{h+1}, after the last round, this prevents an adversary from taking ciphertext and undoing the final substitution and permutation operations.
- The internals of the cipher are protected by k_1 and k_{h+1}. This is called *whitening*.
- Here is a description of *encryption*:

  $A \leftarrow$ *plaintext*
  for $i = 1 \ldots h$ do
  $A \leftarrow A \oplus k_i$ (XOR)
  $A \leftarrow S(A)$ (Substitution)
  $A \leftarrow P(A)$ (Permutation)
  $A \leftarrow A \oplus k_{h+1}$
  ciphertext $\leftarrow A$

- *Decryption* is just the reverse of encryption.
  (The S-box must be invertible!)

AES

- AES is an SPN where the permutation operation is replaced by two linear transformations (one of which is a permutation).
- All operations are *byte* oriented (e.g., S-box maps 8-bits to 8-bits). This allows AES to be efficiently implemented on various platforms.
- The block size of AES is 128 bits.
- Each round key is 128 bits.
- AES accepts three different key lengths. The number of rounds depends on the key length:

key length	h
128	10
192	12
256	14

AES Round Operations

- Each round updates a variable called State which consists of a 4×4 array of bytes (note: $4 \cdot 4 \cdot 8 = 128$, the block size). State is initialized with the plaintext:

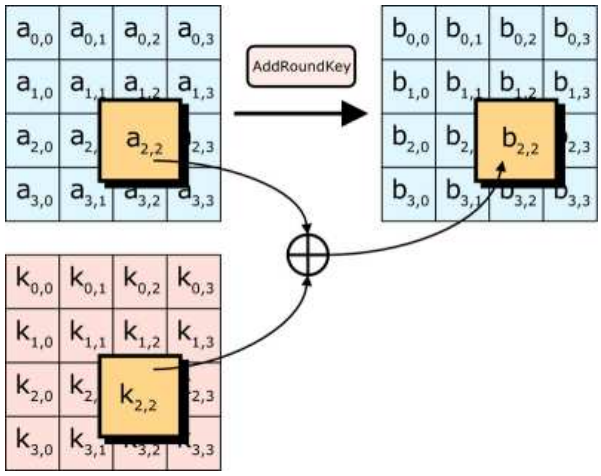
$a_{0,0}$	$a_{0,1}$	$a_{0,2}$	$a_{0,3}$
$a_{1,0}$	$a_{1,1}$	$a_{1,2}$	$a_{1,3}$
$a_{2,0}$	$a_{2,1}$	$a_{2,2}$	$a_{2,3}$
$a_{3,0}$	$a_{3,1}$	$a_{3,2}$	$a_{3,3}$

$\longleftarrow$ *plaintext*

- After h rounds are completed, a final round key is XOR-ed with State, the result being the ciphertext.
- The AES round function uses four operations: `AddRoundKey`, `SubBytes`, `ShiftRows`, `MixColumns`.
- Some arithmetic operations used in AES are carried out in the finite field GF(2^8) (we will not cover finite field arithmetic in this course).

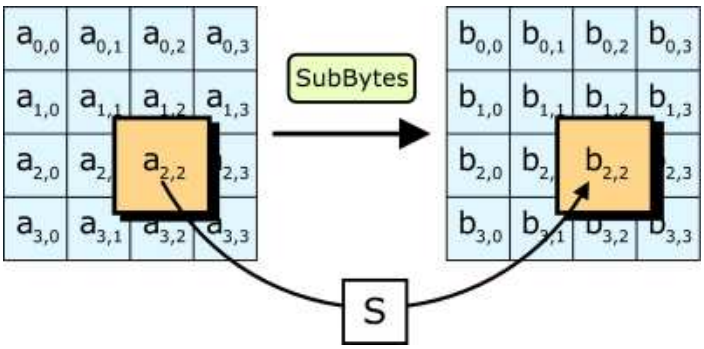
Add Round Key

XOR each byte of State with the corresponding byte of the round key.



Substitute Bytes

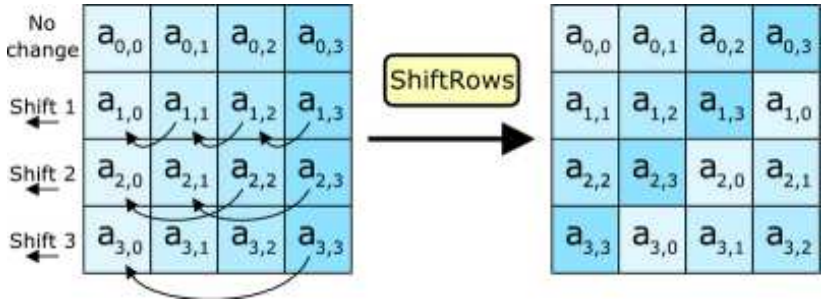
Take each byte in State and replace it with the output of the S-box.



$S : \{0,1\} \rightarrow \{0,1\}^8$ is a fixed and public function.

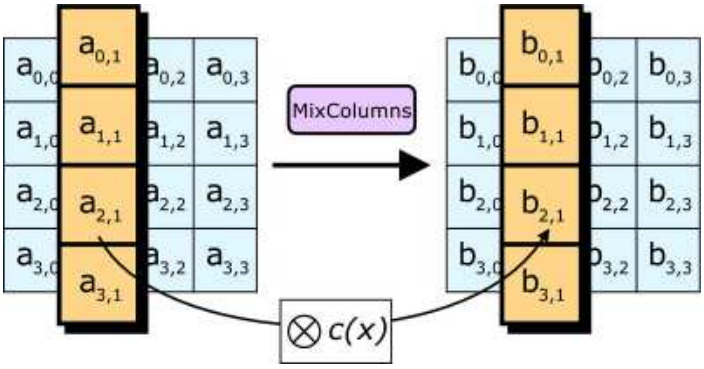
Shift Rows

Permute the bytes of State by applying a *cyclic shift* to each row.



Mix Columns

- Read column i of State as a polynomial: $a_{0,i} + a_{1,i}x + a_{2,i}x^2 + a_{3,i}x^3$
- Multiply this polynomial with the constant polynomial $c(x) = 03 \cdot x^2 + 01 \cdot x^2 + 01 \cdot x + 02$ and reduce modulo $x^4 - 1$. This gives a new polynomial: $b_{0,i} + b_{1,i}x + b_{2,i}x^2 + b_{3,i}x^3$



AES Encryption

- From the key k derive $h+1$ round keys $k_0, k_1, \ldots, k_h$.
- Here is a description of *encryption*:

  State $\leftarrow$ *plaintext*
  State $\leftarrow$ State $\oplus\, k_0$
  for $i = 1 \ldots h-1$ do
    State $\leftarrow$ `SubBytes`(State)
    State $\leftarrow$ `ShiftRows`(State)
    State $\leftarrow$ `MixColumns`(State)
    State $\leftarrow$ State $\oplus\, k_i$
  State $\leftarrow$ `SubBytes`(State)
  State $\leftarrow$ `ShiftRows`(State)
  State $\leftarrow$ State $\oplus\, k_h$
  ciphertext $\leftarrow$ State

- Note that in the final round, `MixColumns` is not applied. This helps make decryption more similar to encryption, and thus is useful when implementing AES. [Details omitted]