41900 – Fundamentals of Security

Symmetric-Encryption & Key Management

Ashish Nanda

Ashish.Nanda@uts.edu.au

A Brief History of Encryption Standards

Year	Major Milestone
1970	**IBM** Research team led by Feistel **develops the LUCIFER cipher** (128-bit blocks and keys).
1973	NBS (now **NIST**) asks for a **proposed data encryption standard.**
1974	**IBM develops DES from LUCIFER.**
1975	**NSA “fixes” DES**: shortens key from 64 to 56 bits, and modifies some S-boxes (substitution boxes).
1977	**DES adopted as a standard.**
1991	**Biham and Shamir discover differential cryptanalysis**, apply their new technique to DES, Find that the **NSA’s modifications had improved security.**
1993	**Michael Wiener** of Nortel **theorizes a USD$1M machine could crack DES in 3.5 hours** using general purpose hardware.
1997	**DES cracked** by brute force **by distributed.net in 96 days**. **NIST** asks for a **proposal for AES** (Advanced Encryption Standard).
1999	**DES cracked in 24 hours** by **distributed.net** and the **EFF USD$250,000 Deep Crack machine**
2000	**Rijndael accepted as AES** (128/192/256-bit key space, 128-bit blocks)

Data Encryption Standard (DES)

DES

DES is a block cipher operating on 64-bit blocks, using a 56-bit key.

- Developed in the early 1970's at IBM.
- "Tweaked" by the NSA (National Security Agency) before release in 1977.
- The world's most heavily analyzed and used cipher.

The NSA's modifications to DES were thought to be adding a "back door".

- Differential Cryptanalysis (DC) had been discovered by IBM in the 1970s (and used in the construction of DES), but IBM were gagged by the NSA.
- The NSA had used DC to strengthen DES, while no-one else was aware it existed.

Attacks on DES

Exhaustive Key Search

- For any **n**-bit block cipher, **j**-bit key, the key can be recovered on average in $\mathbf{2^{j-1}}$ operations, given a small number **(< (j + 4)/n)** of plaintext/ciphertext pairs

- For **DES**, **j = 56**, **n = 64** so exhaustive key search is expected to yield the key in $\mathbf{2^{55}}$ operations.

2DES

Double Encryption with DES (2DES) uses two encryption keys:

$$\mathbf{2DES_{K1,K2}(m) = E_{K1}(E_{K2}(m))}$$

2DES is bad

- 2DES is vulnerable to meet-in-the-middle attack with known plaintext

What does this mean?

- 2DES can be broken in 2^{56} operations on average, using 2^{56} memory slots. (A time-space trade-off!).
- This is not good when there should be 112-bits (56 + 56) of key.

3DES

Two-key Triple DES (3DES) uses DES 3 times using 2 keys. (*112 bits*)

$$\mathbf{3DES_{K1,K2}(m) = E_{K1}(E_{K2}(E_{K1}(m)))}$$

Three-key Triple DES (3DES) uses DES 3 times using 3 keys. (*112 bits*)

$$\mathbf{3DES_{K1,K2,K3}(m) = E_{K1}(E_{K2}(E_{K3}(m)))}$$

DESX

A modification of DES to avoid exhaustive key search is **DESX**.

K1 = 56bits (DES Key)

K2 = 64bits (Whitening Key)

K3 = 64bits hash(K_2, K_3)

$$\mathbf{DESX_{K1,K2,K3}(m) = K_3 \oplus E_{K1}(m \oplus K_2)}$$

The **whitening key** gives greater resilience to brute force attacks.

Advanced Encryption Standard (AES)

AES

In 1997 NIST announced that a competition would be held to choose a new cipher to replace the outdated DES cipher, this to be was named the Advanced Encryption Standard – AES.

Criteria:

- Strength $\geqslant$ 3DES, but much better efficiency
- Flexible - can be implemented in software, hardware or smartcards
- Simple and Elegant
- Block cipher : 128 bit blocks
- 128/192/256 bit keys
- Royalty-free worldwide
- Security for over 30 years
- May protect sensitive data for over 100 years
- Public confidence in the cipher

AES Candidates

15 submissions from the international field.

A number of strong schemes were shortlisted

Name	Type	Rounds	Rel. Speed (cycles)	Gates
Twofsh	Feistel	16	1254	23k
Serpent	SP-network	32	1800	70k
Mars	Type-3 Feistel	32	1600	70k
Rijndael	SP-network	10, 12, 14	1276	-
RC6	Feistel	20	1436	-

AES Finalist

Rijndael (pronounced [rϵinda:l] “rain-dahl”) announced October 2000

- Operates on 128 bit blocks
- Key length is variable: 128, 192 or 256 bits
- It is an SP-network (substitution-permutation network)
- Uses a single S-box which acts on a byte input to give a byte output (a 256 byte lookup table):

$$\mathbf{S(x) = M(x^{-1}) + b \text{ over } GF(2^8)}$$

Where **M** is a predefined matrix, **b** is a constant and **GF** is chosen Galois Feld (nonlinearity comes from $\mathbf{x \rightarrow x^{-1}}$).

- Construction gives tight differential and linear bounds

AES Overview - Rounds

The number of rounds are variable:

- 10 rounds – 128 bit keys
- 12 rounds – 192 bit keys
- 14 rounds – 256 bit keys

Rounds have a 50% margin of safety based on current known attacks.

Potential attacks (which require an enormous number of plaintext/ciphertext pairs) are possible on:

- Only 6 rounds for 128 bit keys
- Only 7 rounds for 192 bit keys
- Only 9 rounds for 256 bit keys

Safety against possible attacks believed to currently be ≈ 100%

Key Distribution

Definitions

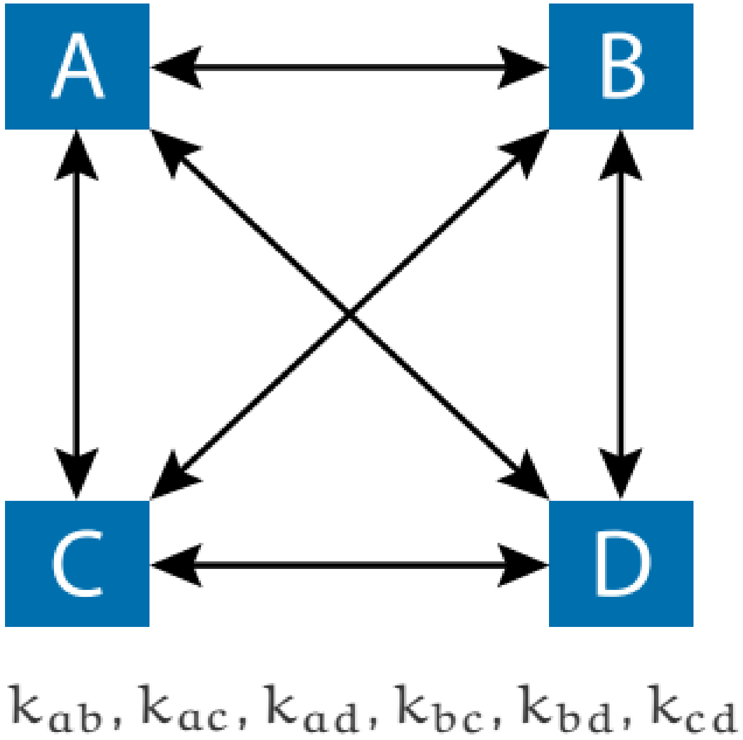
Key Establishment

- The process whereby a shared key becomes available to two or more parties for subsequent cryptographic use.

Key Management

- The set of processes and mechanisms which support key establishment and the maintenance of on going key relationships between parties, including replacing older keys with newer ones.
- Includes:
  - Key agreement
  - Key transport

Key Management



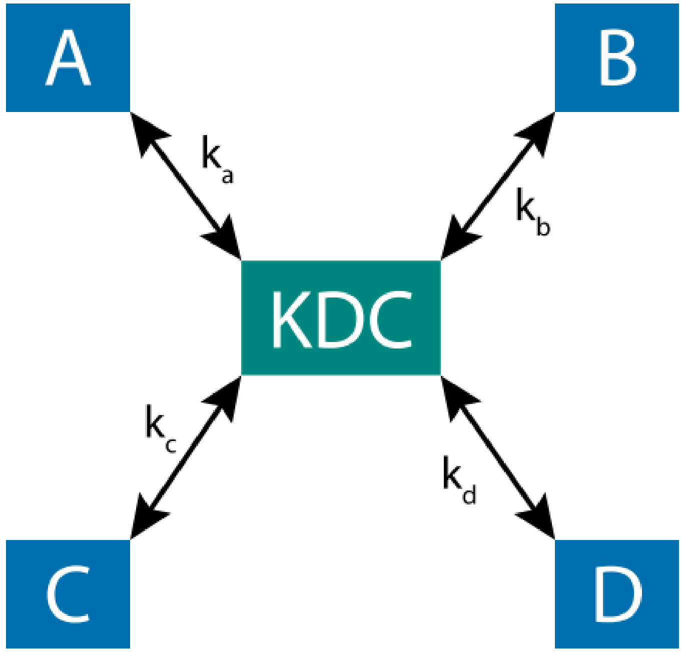
Suppose we have a symmetric key network where Alice, Bob, Carol and Dave want to talk to each other.

For secure communication with **n** parties, we require:

$$\frac{\mathbf{n(n-1)}}{\mathbf{2}}\ \mathbf{keys}$$

Key distribution and management becomes a major issue!

Key Distribution Centre: Naïve



A Key Distribution Centre

Alice →KDC

- I want to talk to Bob

KDC →Alice

- KDC chooses random $\mathbf{K_{AB}}$
- Returns: $\mathbf{E_{KA}(K_{AB}), E_{KB}(K_{AB}},$ 'for talking to Alice')

Alice decrypts $E_{KA}(K_{AB})$ to get K_{AB}

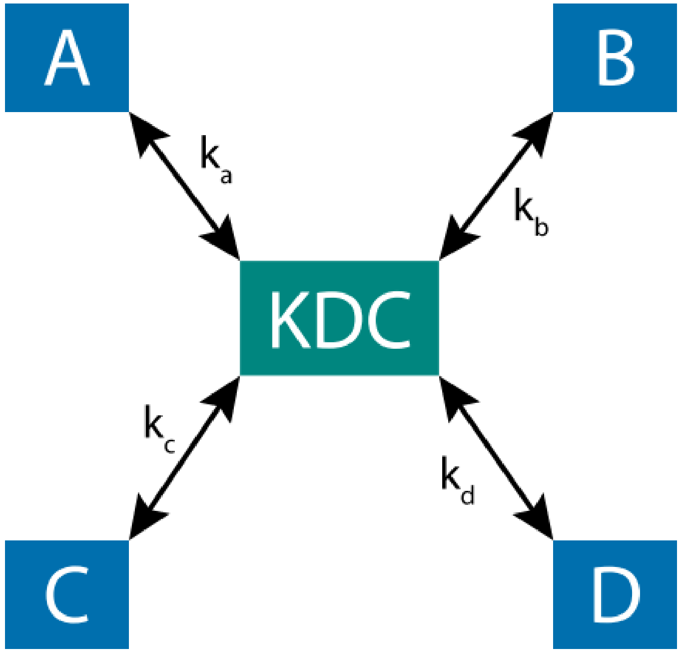
Alice →Bob

- $\mathbf{E_{KB}(K_{AB}},$ 'for talking to Alice')

Bob decrypts using K_B to get K_{AB}

Alice & Bob now share K_{AB}

Key Distribution Centre: Naïve



A Key Distribution Centre

Problems:

- The Key Distribution Centre is a single point of failure – *likely to be attacked*
- No authentication
- Poor scalability
- Slow

Merkle’s Puzzles

Merkle’s Puzzles are a way of doing key exchange between Alice and Bob without the need for a third party.

Alice creates **N** puzzles $\mathbf{P_1, P_2, \ldots, P_N}$, of the form

$\mathbf{P_i = E_{pi}}$(“This is puzzle **#X_i**”, $\mathbf{k_i}$)

- **N ≈ 200**
- **$|P_i|$ ≈ 20 bits (weak)**
- **$|k_i|$ ≈ 128 bits (strong)**
- $\mathbf{X_i}$**,** $\mathbf{p_i}$**,** and $\mathbf{k_i}$ are chosen randomly and different for each **i**.

Merkle’s Puzzles

Alice sends all puzzles to Bob: $\mathbf{P_1, P_2, . . . , P_N}$.

Bob chooses a random puzzle $\mathbf{P_j}$ for some $\mathbf{j \in \{1, 2, . . . , N\}}$.

- Finds $\mathbf{p_j}$ by brute force (key space search)
- Recovers $\mathbf{k_j}$ and $\mathbf{X_j}$
- Bob sends $\mathbf{X_j}$ to Alice unencrypted

Alice looks up the index of $\mathbf{X_j}$ to find the key $\mathbf{k_j}$ chosen by Bob.

Alice & Bob both share key $\mathbf{k_j}$

Attacking Merkle's Puzzles

On average, Eve must break half of the puzzles to find which puzzle contains $\mathbf{X_j}$ (and hence obtain $\mathbf{k_j}$).

So for $\mathbf{2^{20}}$ puzzles, Eve must try $\mathbf{2^{19}}$ puzzles on average.

Each puzzle is encrypted with the 20 bit key $\mathbf{p_i}$. Eve must search, on average, half of the key space:

$$\mathbf{2^{19}.\ 2^{19} \times 2^{19} = 2^{38}}$$

Attacking Merkle's Puzzles

If Alice and Bob can try 10,000 keys per second:

- It will take about 1 minute for each to perform their steps

  Alice to generate, and Bob to break $\mathbf{p_j = 2^{19}}$ keys
- Plus another minute to communicate all the puzzles over ADSL

With comparable resources, it will take Eve about a year to break the system.

Note: Merkle's puzzles uses a lot of bandwidth – impractical!

Diffie-Hellman Key Exchange

Diffie-Hellman key exchange (Stanford, 1976) is a protocol for establishing a cryptographic key using mathematical tricks. It is a worldwide standard for use in SSL, smartcards, etc.

The rough idea is this: (details later)

- Alice and Bob agree on some number **g**.
- Alice generates a random number **a**, and sends $\mathbf{g^a}$ to Bob.
- Bob generates a random number **b**, and sends $\mathbf{g^b}$ to Alice.
- Alice and Bob can each compute $\mathbf{g^{ab}}$, their shared secret.

An eavesdropper only has $\mathbf{g^a}$, $\mathbf{g^b}$, and **g**. Assuming that calculating logarithms is hard, they cannot recover **a** or **b**.

Diffie-Hellman Key Exchange