Module 1: Cryptography I

Nitesh Saxena

Overview of the Module

Module 1, Lecture 1

Cryptography Overview

Cryptography

- Etymology: Secret (Crypt) Writing (Graphy)
- Study of mathematical techniques to achieve various goals in information security, such as confidentiality, authentication, integrity, non-repudiation, etc.
- Not the only means of providing network security, rather a subset of techniques.
- Quite an old field!

Cryptography: Cast of Characters

- Alice (A) and Bob (B): communicating parties
- Eve (E): Eavesdropping (or **passive**) adversary
- Mallory (M): Man-in-the-Middle (or **active adversary)**
- Trent (T): a trusted third party (TTP)

Focus of Module 1

- How to achieve confidentiality by means of cryptography?

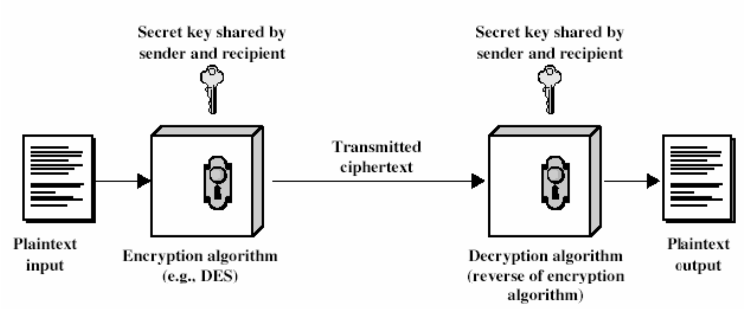
Private Key/Public Key Cryptography

- **Private Key**: Sender and receiver share a common (private) key
  - Encryption and Decryption is done using the private key
  - Also called conventional/shared-key/single-key/ symmetric-key cryptography
- **Public Key**: Every user has a private key and a public key
  - Encryption is done using the public key and Decryption using private key
  - Also called two-key/asymmetric-key cryptography

Common Terminologies

- Plaintext
- Key
- Encrypt (encipher)
- Ciphertext
- Decrypt (decipher)
- Cipher
- Cryptosystem
- Cryptanalysis (codebreaking)
- Cryptology: Cryptography + Cryptanalysis

Private key model



Open vs Closed Design

- Closed Design (as was followed in military communication during the World Wars)
  - Keep the cipher secret
  - Also sometimes referred to as the “proprietary design”
  - Bad practice! (why?)
- Open Design (*Kerckhoffs' principle*)
  - Keep everything public, except the key
  - Good practice – this is what we focus upon!

Module 1, Lecture 2

Private Key Encryption

Private Key Encryption: main functions

1. KeyGen: K = KeyGen(l) (l is a security parameter)
2. Enc: C = Enc(K,M)
3. Dec: M = Dec(K,C)

Goals of the Attacker

- Learn the plaintext corresponding to a given ciphertext -- **One-Way Security**
- Extract the key – **Key Recovery Security**
- Learn some information about the plaintext corresponding to a given ciphertext – **Semantic Security**
- *Key recovery security and one-way security are a must for an encryption scheme. Semantic Security is ideal.*

Capabilities of the Attacker

1. **No Information** (besides the algorithm)
2. **Ciphertext only**
   – Adversary knows only the ciphertext(s)
3. **Known plaintext**
   – Adversary knows a set of plaintext-ciphertext pairs
4. **Chosen (and adaptively chosen) plaintext (CPA attack)**
   – Adversary chooses a number of plaintexts and obtains the corresponding ciphertexts
5. **Chosen (and adaptively chosen) ciphertext attack (CCA attack)**
   – Adversary chooses a number of ciphertexts and obtains the corresponding plaintexts

Security Model

least attacker capability .... most attacker capability

1<2<3<4<5

weakest cryptosystem .... strongest cryptosystem

- 1 is the hardest and 5 is the easiest attack to perform
- A cryptosystem secure against 5 is the strongest, and secure against 1 is the weakest
- A cryptosystem secure against 5 is automatically secure against 4, 3, 2 and 1

Brute Force Attacks: Key Recovery

- Since the key space is finite, given a pair (or more) of plaintext and ciphertext, a cryptanalyst can try and check all possible keys.
- For above to be not feasible, key space should be large!!
  - How large?
  - Large enough to make it impractical for an adversary. But what is impractical today, may not be so tomorrow. At least 2^{80} – see this paper on "selecting cryptographic key sizes"
    - https://infoscience.epfl.ch/record/164526/files/NPDF-22.pdf

Module 1, Lecture 3

Classical Ciphers

Classical Ciphers

- Substitution Ciphers
- Transposition Ciphers
- Examples: Caesar’s Cipher, Vigenere Cipher
- All of these are insecure due to language characteristic analysis

One Time Pad or Vernam Cipher

- plaintext is binary string and key is binary string of equal length, then encryption can be done by a simple XOR operation.

  Plaintext: 01010000010001010011

  Key: 11010101001001100111

  Ciphertext: 10000101011000110100
- **If the key is random** and **is not re-used,** then such a system offers unconditional security – perfect secrecy!
- Intuitively **perfect secrecy** can be seen from the fact that given any plaintext and ciphertext, there is a key which maps the selected plaintext to the selected ciphertext. So given a ciphertext, we get no information whatsoever on what key or plaintext could have been used.
- How do we obtain "random" bit-strings for shared secret keys as long as the messages, and never re-use them?
- System is **not practical**.

Module 1, Lecture 4

DES Functioning

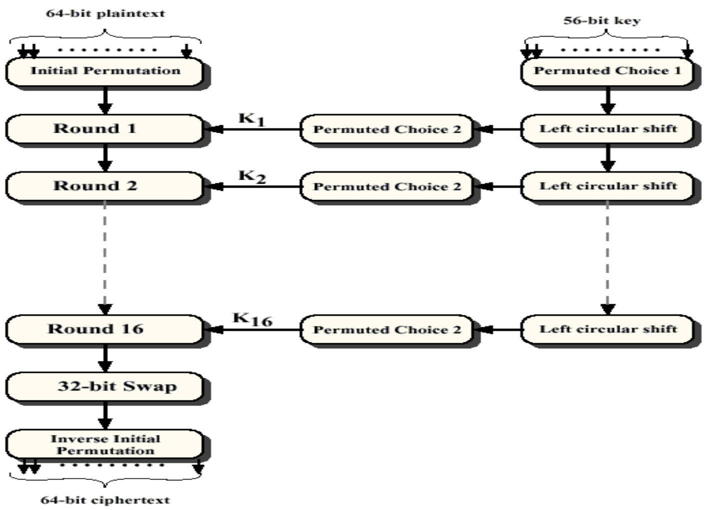
Block Ciphers and Stream Ciphers

- Block ciphers partition plaintext into blocks and encrypt each block independently (with the same key) to produce ciphertext blocks.
- A stream cipher generates a *keystream* and encrypts by combining the keystream with the plaintext, usually with the bitwise XOR operation.
- We will focus mostly on Block Ciphers

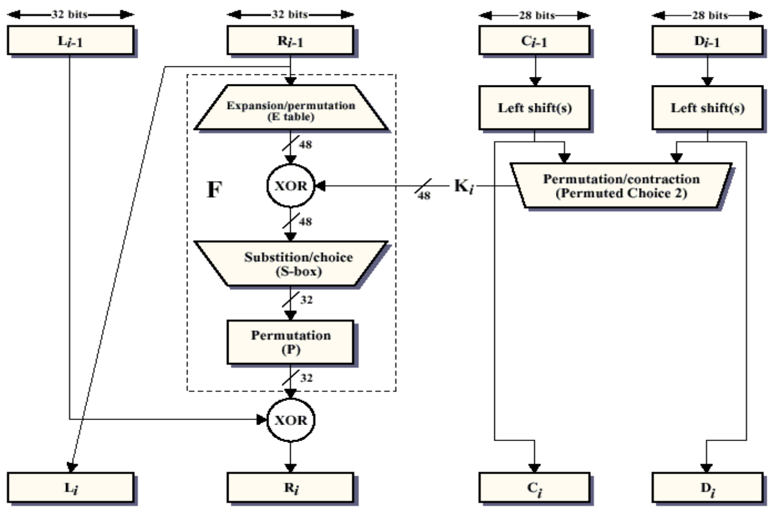
DES – Data Encryption Standard

- Encrypts by series of substitution and transpositions.
- Based on *Feistel Structure*
- Worldwide standard for more than 20 years.
- Designed by IBM (Lucifer) with later help from NSA.
- No longer considered secure for highly sensitive applications.
- Replacement standard AES (advanced encryption standard) recently completed.

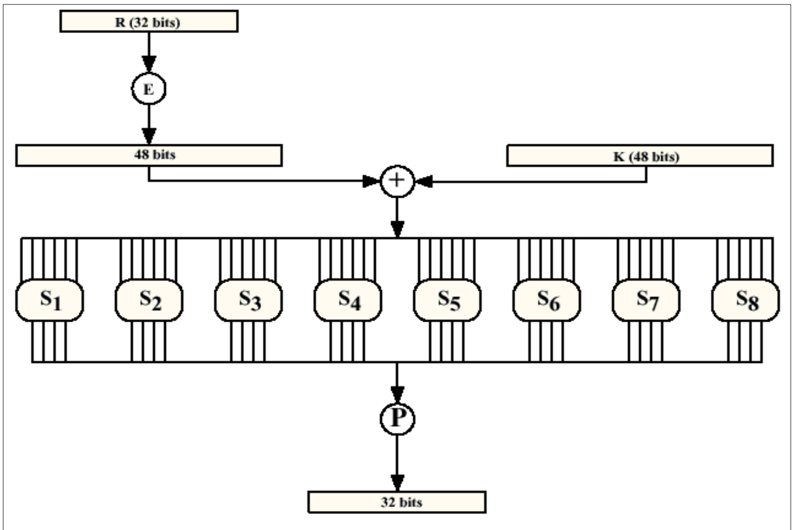
DES – Overview (Block Operation)



DES – Each Round



DES – Function F



DES Decryption

- Same as the encryption algorithm with the "reversed" key schedule

DES Example

We choose a random plaintext block and a random key, and determine what the ciphertext block would be (all in hexadecimal):

Plaintext: 123456ABCD132536 Key: AABB09182736CCDD
CipherText: C0B7A8D05F3A829C

Plaintext: 123456ABCD132536			
After initial permutation: 14A7D67818CA18AD After splitting: L_0=14A7D678 R_0=18CA18AD			
Round	*Left*	*Right*	*Round Key*
Round 1	18CA18AD	5A78E394	194CD072DE8C
Round 2	5A78E394	4A1210F6	4568581ABCCE
Round 3	4A1210F6	B8089591	06EDA4ACF5B5
Round 4	B8089591	236779C2	DA2D032B6EE3

Example (contd) -- encryption

Round 5	236779C2	A15A4B87	69A629FEC913
Round 6	A15A4B87	2E8F9C65	C1948E87475E
Round 7	2E8F9C65	A9FC20A3	708AD2DDB3C0
Round 8	A9FC20A3	308BEE97	34F822F0C66D
Round 9	308BEE97	10AF9D37	84BB4473DCCC
Round 10	10AF9D37	6CA6CB20	02765708B5BF
Round 11	6CA6CB20	FF3C485F	6D5560AF7CA5
Round 12	FF3C485F	22A5963B	C2C1E96A4BF3
Round 13	22A5963B	387CCDAA	99C31397C91F
Round 14	387CCDAA	BD2DD2AB	251B8BC717D0
Round 15	BD2DD2AB	CF26B472	3330C5D9A36D
Round 16	19BA9212	CF26B472	181C5D75C66D
After combination: 19BA9212CF26B472			
Ciphertext: C0B7A8D05F3A829C *(after final permutation)*			

Example (contd) -- decryption

Let us see how Bob, at the destination, can decipher the ciphertext received from Alice using the same key. Table 6.16 shows some interesting points.

Ciphertext: C0B7A8D05F3A829C			
After initial permutation: 19BA9212CF26B472 After splitting: L_0=19BA9212 R_0=CF26B472			
Round	*Left*	*Right*	*Round Key*
Round 1	CF26B472	BD2DD2AB	181C5D75C66D
Round 2	BD2DD2AB	387CCDAA	3330C5D9A36D
. . .	. . .	. . .	. . .
Round 15	5A78E394	18CA18AD	4568581ABCCE
Round 16	14A7D678	18CA18AD	194CD072DE8C
After combination: 14A7D67818CA18AD			
Plaintext:123456ABCD132536 (after final permutation)			

DES Security: Avalanche Effect

Plaintext: 0000000000000000 Key: 22234512987ABB23
Ciphertext: 4789FD476E82A5F1

Plaintext: 000000000000000**1** Key: 22234512987ABB23
Ciphertext: 0A4ED5C15A63FEA3

Although the two plaintext blocks differ only in the rightmost bit, the ciphertext blocks differ in 29 bits. This means that changing approximately 1.5 percent of the plaintext creates a change of approximately 45 percent in the ciphertext.

Further Reading

- Chapter 7.4 of HAC
- Chapter 3 of Stallings

Module 1, Lecture 5

DES Security

DES Security

- S-Box design not well understood
- Has survived some recent sophisticated attacks (differential cryptanalysis)
- Key is too short. Hence is vulnerable to brute force attack.
- 1998 distributed attack took 3 months.
- $1,000,000 machine will crack DES in 35 minutes – 1997 estimate. $10,000 – 2.5 days.

DES Cracking machine

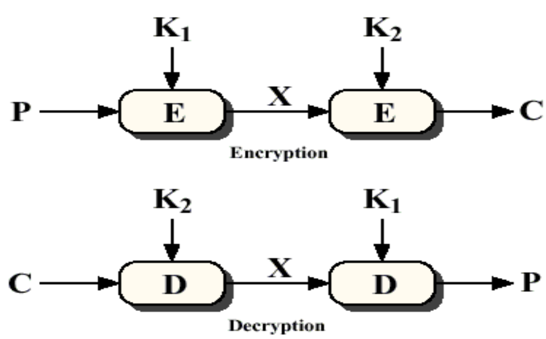
Super-encryption.

- If key length is a concern, then instead of encrypting once, encrypt twice!!

$$C = E_{K2}(E_{K1}(P))$$
$$P = D_{K1}(D_{K2}(C))$$

- Does this result in a larger key space?
- Encrypting with multiple keys is known as super-encryption.
- May not always be a good idea

Double DES



- Double DES is almost as easy to break as single DES (Needs more memory though)!

Double DES – Meet-in-the-middle Attack (due to Diffie-Hellman)

- Based on the observation that, if

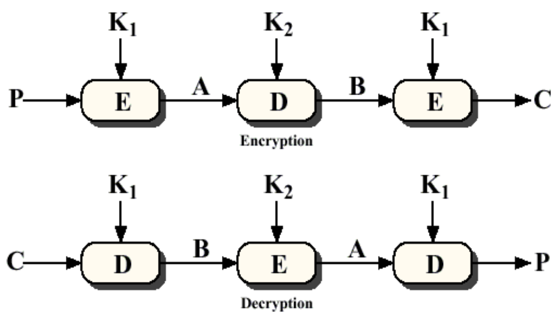
$$C = E_{K2}(E_{K1}(P))$$

Then

$$X = E_{K1}(P) = D_{K2}(C).$$

- Given a known (P, C) pair, encrypt P with all possible values of K and store result in table T.
- Next, decrypt C with all possible keys K and check result. If match occurs then check key pair with new known (P, C) pair. If match occurs, you have found the keys. Else continue as before.
- Process will terminate successfully.

Triple DES



- Triple DES (2 keys) requires 2^{112} search. Is reasonably secure.
- Triple DES (3 keys) requires 2^{112} as well
- Which one is better?