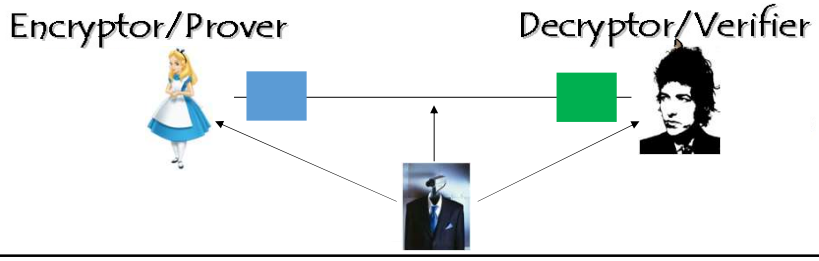
Crypto Basics

Crypto Attacks:
- ciphertext only
- known plaintext
- chosen plaintext
- chosen ciphertext
- brute force

Cryptosystem:
- *P -- plaintext*
- *C -- ciphertext*
- *K -- keyspace*
- *E -- encryption rules*
- *D -- decryption rules*



1

1

Average Time for Exhaustive Key Search (for Brute-Force Attacks)

Key Size (bits)	**Number of Alternative Keys**	**Time required at 10^6 Decr/µs**
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

Today, **> 80 bits** can be generally considered "short-term secure"
(but things are changing rapidly due to GPUs and special hardware; better treat this only as the minimum key-size assuming small-time attackers)

2

2

Types of Attainable Security

• Perfect, unconditional or "information theoretic": the security is evident free of any (computational/hardness) assumptions

• Reducible: security can be shown to be based on some common (often unproven) assumptions, e.g., the conjectured difficulty of factoring large integers

• Ad hoc: the security seems good → "snake oil"…

Take a look at:

http://www.ciphersbyritter.com/GLOSSARY.HTM

3

3

Computational Security

- Encryption scheme is *computationally secure* if
  - cost of breaking it (via brute force) exceeds the value of the encrypted information; or
  - time required to break it exceeds useful lifetime of the encrypted information
- Most modern schemes we will see are considered computationally secure
  - Usually rely on having a very large key-space, impregnable to brute force attacks
- Most advanced schemes rely on lack of knowledge of effective algorithms for certain hard problems, not on a proven inexistence of such algorithms (reducible security)!
  - Such as: factorization, discrete logarithms, etc.

4

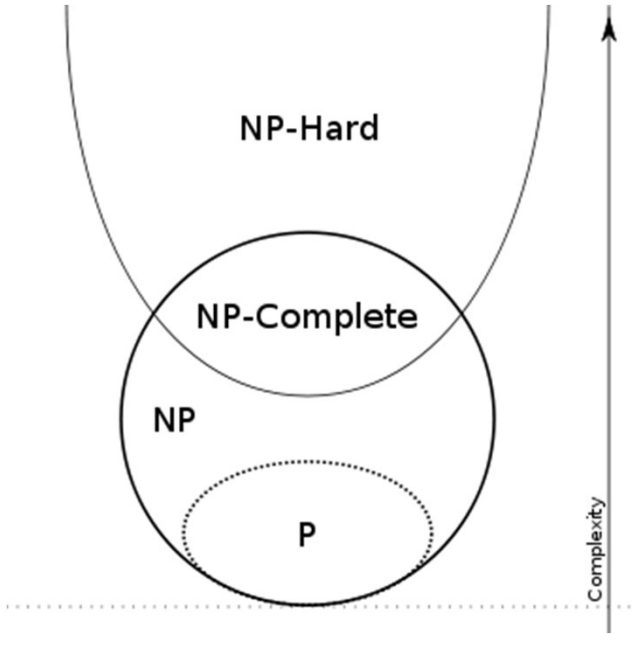
4

Complexity Reminder/Re-cap

- **P**: problems that can be solved in polynomial time, i.e., problems that can be solved/decided "efficiently"
- **NP**: broad set of problems that includes P;
  - answers can be verified "efficiently" (in polynomial time);
  - solutions cannot always be efficiently found (as far as we know).
- **NP-complete**: believed-to-be-hard decision problems in NP; they appear to have no efficient solution; answers are efficiently verifiable, solution to one is never much harder than a solution to another
- **NP-hard**: hardest; some of them may not be solved by a non-deterministic TM. Many computational version of NP-complete problems are NP-hard.

5

5



6

6

Complexity Reminder/Re-cap

- **P**: problems that can be solved in polynomial time, i.e., problems that can be solved/decided "efficiently"
- **NP**: broad set of problems that includes P;
  - answers can be verified "efficiently" (in polynomial time);
  - solutions cannot always be efficiently found (as far as we know).
- **NP-complete**: believed-to-be-hard decision problems in NP; they appear to have no efficient solution; answers are efficiently verifiable, solution to one is never much harder than a solution to another
- **NP-hard**: hardest; some of them may not be so
  TM. Many computational version of NP-complet
- **Examples:**
  - Factoring, discrete log are in NP, not known if NP-complete or P
  - Knapsack is NP-complete
  - Primality testing was "recently" (2002) shown to be in P

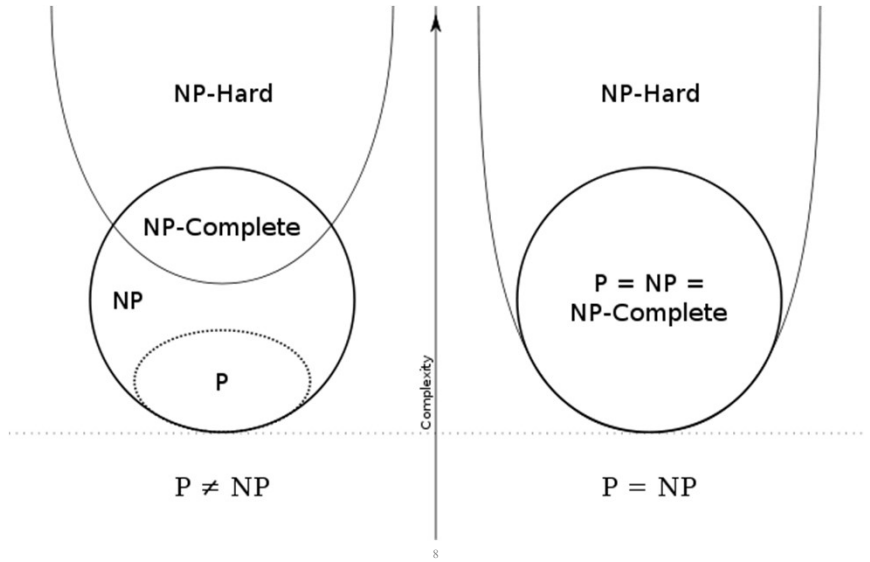
Relied upon by today's encryption schemes!

For more info, see: **https://www.nist.gov/dads//**

7

7

P vs NP



8

Suggested Readings:

Chapters 1 and 2 in KPS book

Optional: Ch 1 in Stinson

9

9

Lecture 3

Encryption I

Suggested Readings:

- Chs 3 & 4 in KPS (recommended)
- Ch 3 in Stinson (optional)

[lecture slides are adapted from previous slides by Prof. Gene Tsudik]

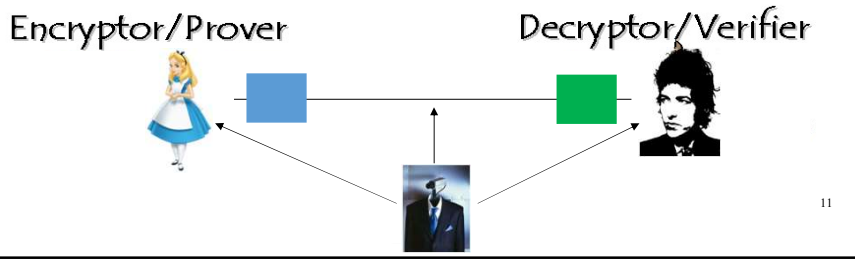
10

10

Crypto Basics

Crypto Attacks:
- ciphertext only
- known plaintext
- chosen plaintext
- chosen ciphertext
- brute force

Cryptosystem:
- *P -- plaintext*
- *C -- ciphertext*
- *K -- keyspace*
- *E -- encryption rules*
- *D -- decryption rules*



11

11

Cryptosystems

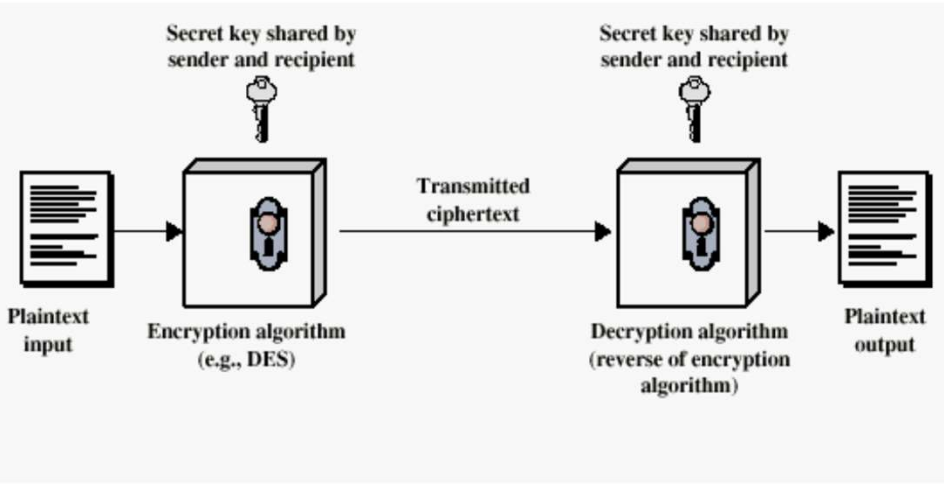
Classified along three dimensions:

- Type of operations used for transforming plaintext into ciphertext
  - Binary arithmetic: shifts, XORs, ANDs, etc.
    - Typical for **conventional/symmetric** encryption
  - Integer arithmetic
    - Typical for **public key/asymmetric** encryption
- Number of keys used
  - Symmetric or conventional (single key shared)
  - Asymmetric or public-key (2 keys: 1 to encrypt, 1 to decrypt)
- How plaintext is processed:
  - One bit at a time – "stream cipher"
  - A block of bits – "block cipher"

12

12

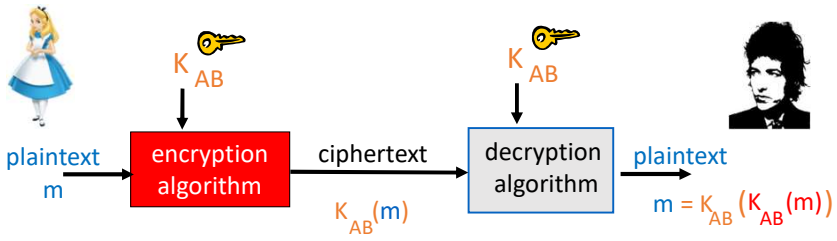
Conventional/Symmetric Encryption Principles



13

13

Conventional/Symmetric Cryptography



- Alice and Bob share a key K_{AB} which they somehow agree upon (how?)
  - key distribution / key management problem
  - ciphertext is roughly as long as plaintext
  - examples: Substitution, Vernam OTP, DES, AES

14

14

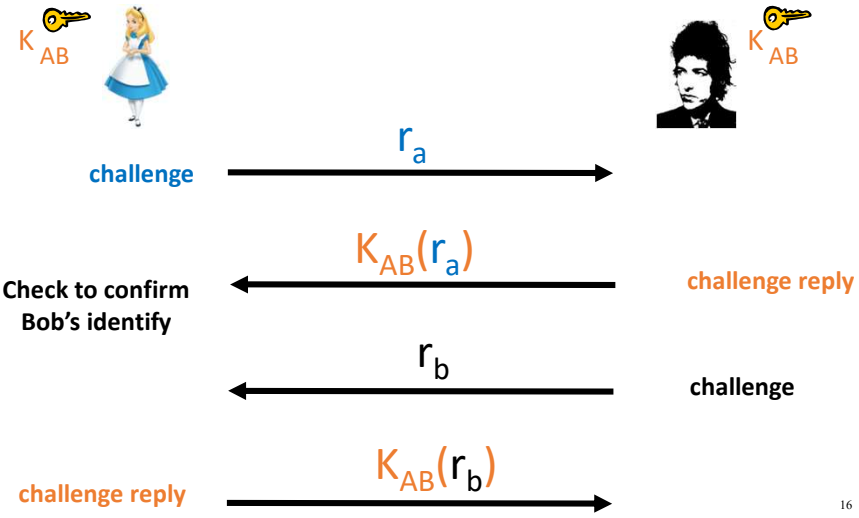
Uses of Conventional/Symmetric Cryptography

- Message transmission (confidentiality):
  - Communication over insecure channels
- Secure storage: encrypted disk/volume on Windows, IOS
- Strong authentication: proving knowledge of a secret without revealing it:

15

15

Challenge-Response Authentication Example



16

16

Uses of Conventional/Symmetric Cryptography

- Message transmission (confidentiality):
  - Communication over insecure channels
- Secure storage: crypt on Unix
- Strong authentication: proving knowledge of a secret without revealing it:
  - Eve can obtain <plaintext, ciphertext> pairs (known plaintext attack)
  - Challenge should be chosen from a large pool
- Integrity checking: fixed-length checksum for message via symmetric key cryptography
  - Send MAC along with the message MAC=H(K, m)

17

17

Conventional/Symmetric Cryptography

- Advantages
  - high data throughput
  - relatively short key size
  - primitives to construct various cryptographic mechanisms
- Disadvantages
  - key must remain secret at both ends
  - key must be distributed securely and efficiently
  - relatively short key lifetime

18

18

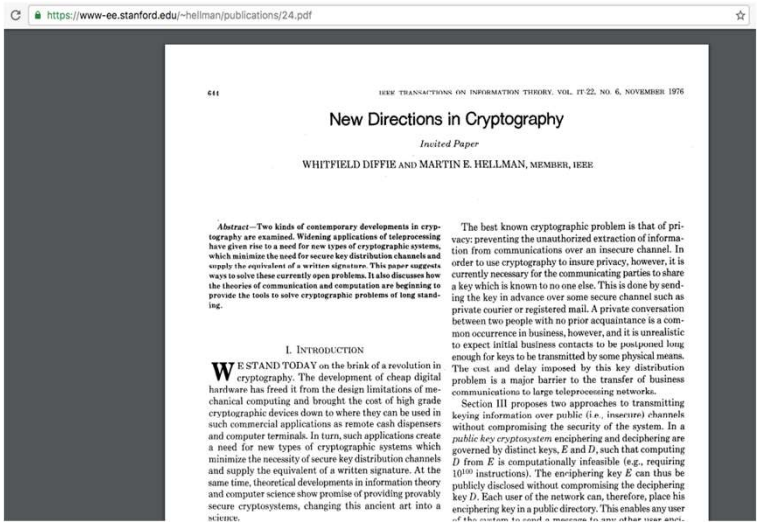
Public Key (Asymmetric) Cryptography

- Invented in 1974-1978: Merkle, Diffie-Hellman (DH), ElGamal, Rivest-Shamir-Adleman (RSA)
  - DH and RSA won Turing awards (2002, 2015)
- Two keys: private (SK), public (PK)
  - Encryption: with the other end's public key;
  - Decryption: with my own private key
  - Digital Signatures: Signing by private key; Verification by public key. i.e., "encrypt" message digest/hash -- $h(m)$ -- with private key
    - Authorship (authentication)
    - Integrity: Similar to MAC
    - Non-repudiation: **can't do with symmetric key cryptography**
- Typically much **slower** (~1000x) than symmetric cryptography
  - Often used together with symmetric cryptography, e.g., to encrypt session keys

19

19

Genesis of Public Key Cryptography: Diffie- Hellman Paper


https://www-ee.stanford.edu/~hellman/publications/24.pdf

644 IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

New Directions in Cryptography

Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

Abstract—Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.

I. INTRODUCTION

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of mechanical computing and brought the cost of high grade cryptographic devices down to where they can be used in such commercial applications as remote cash dispensers and computer terminals. In turn, such applications create a need for new types of cryptographic systems which minimize the necessity of secure key distribution channels and supply the equivalent of a written signature. At the same time, theoretical developments in information theory and computer science show promise of providing provably secure cryptosystems, changing this ancient art into a science.

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

Section III proposes two approaches to transmitting keying information over public (i.e., insecure) channels without compromising the security of the system. In a *public key cryptosystem* enciphering and deciphering are governed by distinct keys, E and D, such that computing D from E is computationally infeasible (e.g., requiring 10^{100} instructions). The enciphering key E can thus be publicly disclosed without compromising the deciphering key D. Each user of the network can, therefore, place his enciphering key in a public directory. This enables any user

20

20

Public Key Cryptography

Bob's public key PK_B

Bob's private key SK_B





21

21

Public Key Cryptography

Bob's public key

PK_B

Bob's private key SK_B



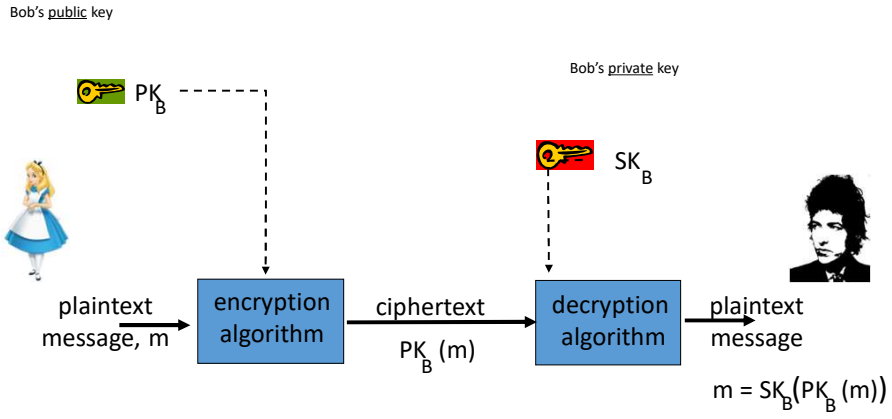


plaintext message, m → encryption algorithm → ciphertext $PK_B(m)$

22

22

Public Key Cryptography



23

23

Uses of Public Key Cryptography

- Data transmission (confidentiality):
  - Alice encrypts m_a using PK_B, Bob decrypts it to obtain m_a using SK_b.
- Secure Storage: encrypt with own public key, later decrypt with own private key
- Authentication:
  - No need to store secrets, only need *public* keys.
  - Secret/symmetric key cryptography: need to share *secret* key for every person you want to communicates with
- Digital Signatures (authentication, integrity, non-repudiation)

24

24

Public Key Cryptography

- Advantages
  - only the private key must be kept secret
  - relatively long life-time of the key
  - more security services
  - relatively efficient digital signature verifications
- Disadvantages
  - low data throughput
  - much larger key sizes
  - distribution/revocation of public keys
  - security based on conjectured hardness of certain computational problems

25

25

Recap: Complexity

- **P:** problems that can be solved in polynomial time, i.e., problems that can be solved/decided "efficiently"
- **NP:** broad set of problems that includes P;
  - answers can be verified "efficiently" (in polynomial time);
  - solutions cannot always be efficiently found (as far as we know).
- **NP-complete:** believed-to-be-hard decision problems in NP; they appear to have no efficient solution; answers are efficiently verifiable, solution to one is never much harder than a solution to another
- **NP-hard:** hardest; some of them may not be so
TM. Many computational version of NP-complet
- **Examples:**
  - Factoring, discrete log are in NP, not known if NP-complete or P
  - Knapsack is NP-complete
  - Primality testing was "recently" (2002) shown to be in P

Relied upon by today's encryption schemes!

For more info, see: **https://www.nist.gov/dads//**

26

26

Comparison Summary

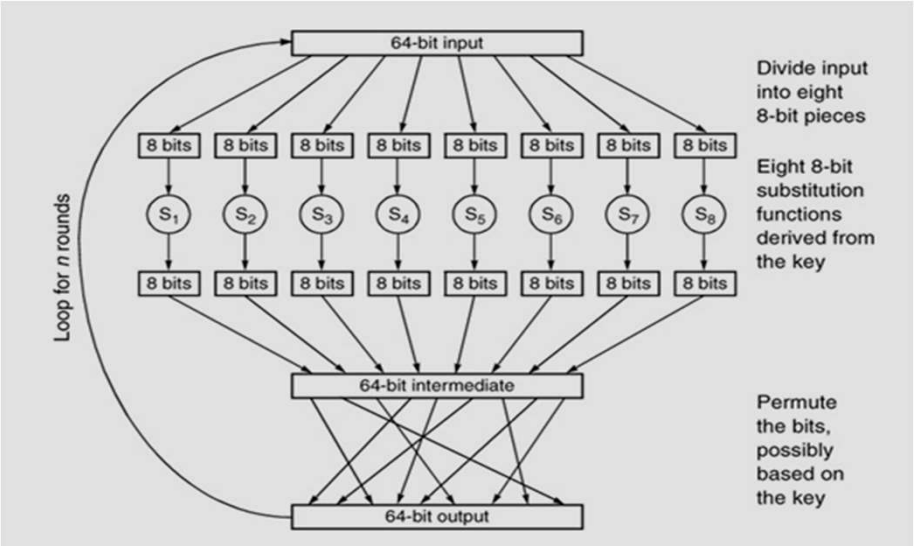
- Public/asymmetric key
  - encryption, signatures (esp., non-repudiation), and key management
- Conventional/symmetric
  - encryption and some data integrity applications
- Key sizes
  - Keys in public key crypto must be larger (e.g., 2048 bits for RSA) than those in conventional crypto (e.g., 112 bits for 3-DES or 256 bits for AES)
    - Most attacks on (otherwise secure) conventional cryptosystems are via exhaustive key search (brute force)
    - Whereas, public key cryptosystems might be subject to "short-cut" attacks (e.g., factoring large composite numbers in RSA)

(Symmetric crypto)

"Modern" Block Ciphers

Data Encryption Standard (DES)

Generic Example of Block Encryption



20

29

Feistel Cipher Structure

- Virtually all conventional block encryption algorithms, including DES, have a structure first described by Horst Feistel of IBM in 1973
- Specific realization of a Feistel Network depends on the choice of the following parameters and features:

30

30

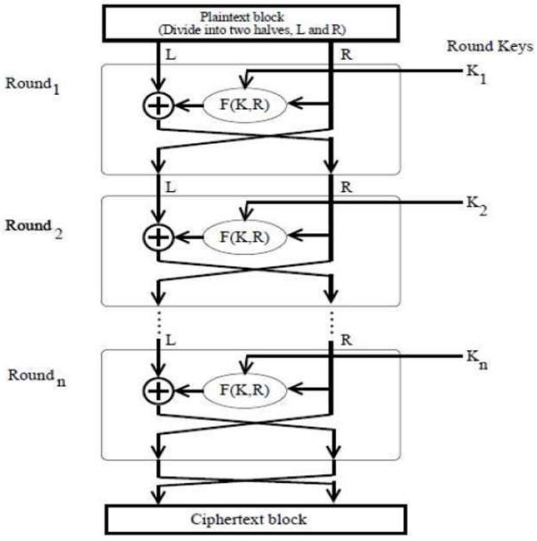
Feistel Cipher Structure

- **Block Size:** larger block sizes mean greater security
- **Key Size:** larger key size means greater security
- **Number of Rounds:** multiple rounds offer increasing security
- **Subkey Generation Algorithm:** greater complexity leads to greater difficulty of cryptanalysis

31

31

Classic Feistel Network



"Round Keys" are generated from original key via subkey generation algorithm

32

32

Block Ciphers

- Originated with early 1970's IBM effort to develop banking security systems
- First result was Lucifer, most common variant has 128-bit key and block size
- Was not secure in any of its variants (later realized)
- Called a Feistel or product cipher
- F()-function is a simple transformation, does not have to be reversible
- Each step is called a round; the more rounds, the greater the security (to a point)
- Most famous example of this design is DES

33

33

Conventional Encryption Standard

Data Encryption Standard (DES)

- Most widely used encryption method in 1970s/80s/90s
  - AES took over in early 2000-s
- Block cipher (in native ECB mode)
- Plaintext processed in 64-bit blocks
- Key is 56 bits

34

34

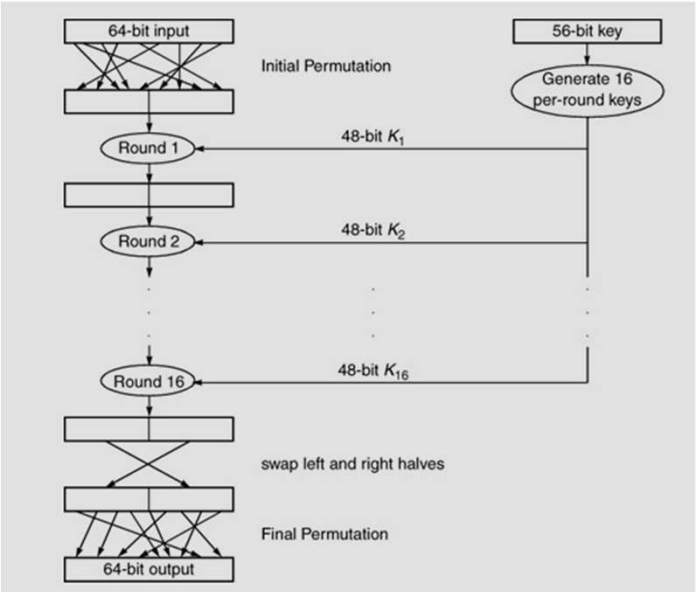
Data Encryption Standard (DES)

- 64-bit input block
- 64-bit output block
- 16 rounds
- 64 (effective 56) bit key
  - 8 least significant bits (1 per byte) reserved for parity (odd)
- Key schedule computed at startup
- Aimed at bulk data
  - > 16 rounds does not help
- Other S-boxes usually hurt …

25

35

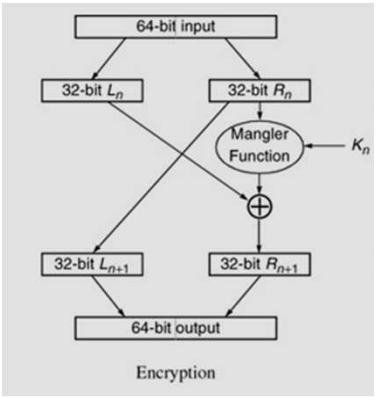
Basic Structure of DES



26

36

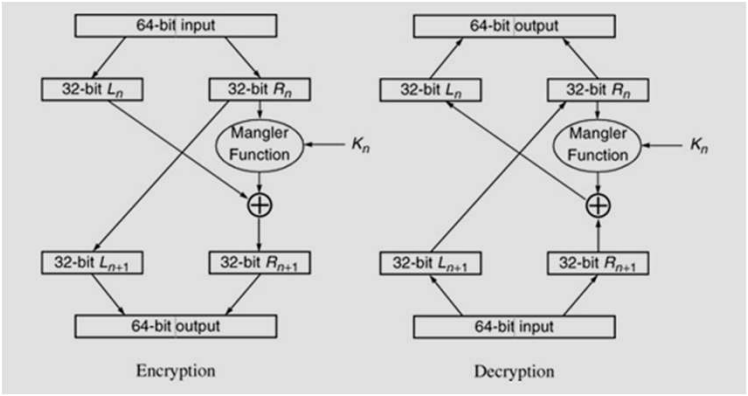
Encryption vs Decryption in DES



37

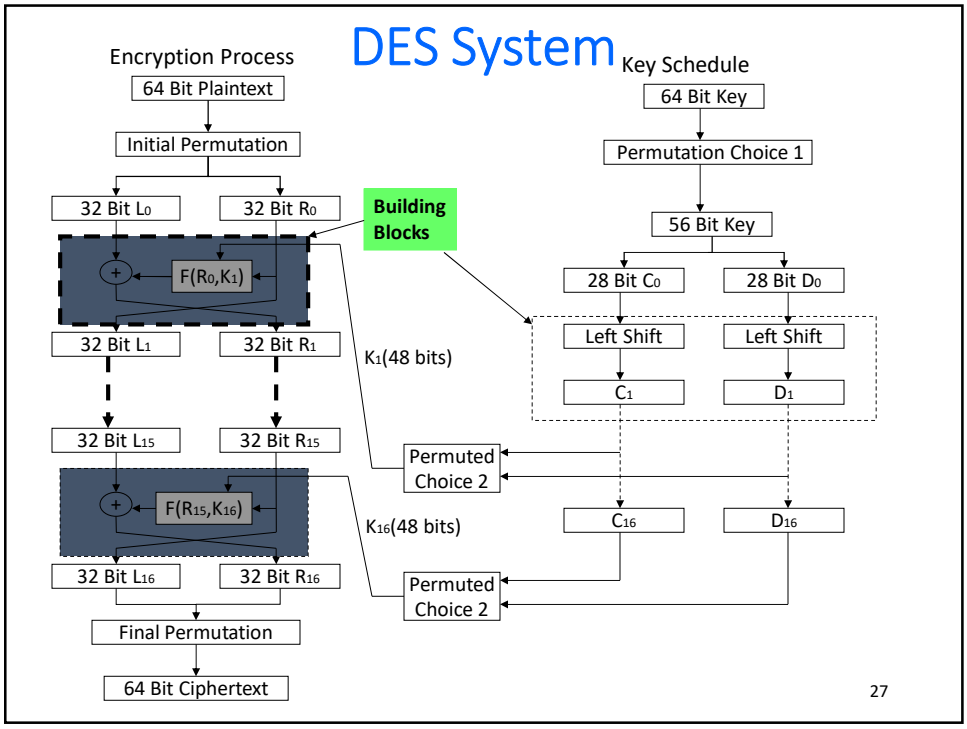
37

Encryption vs Decryption in DES

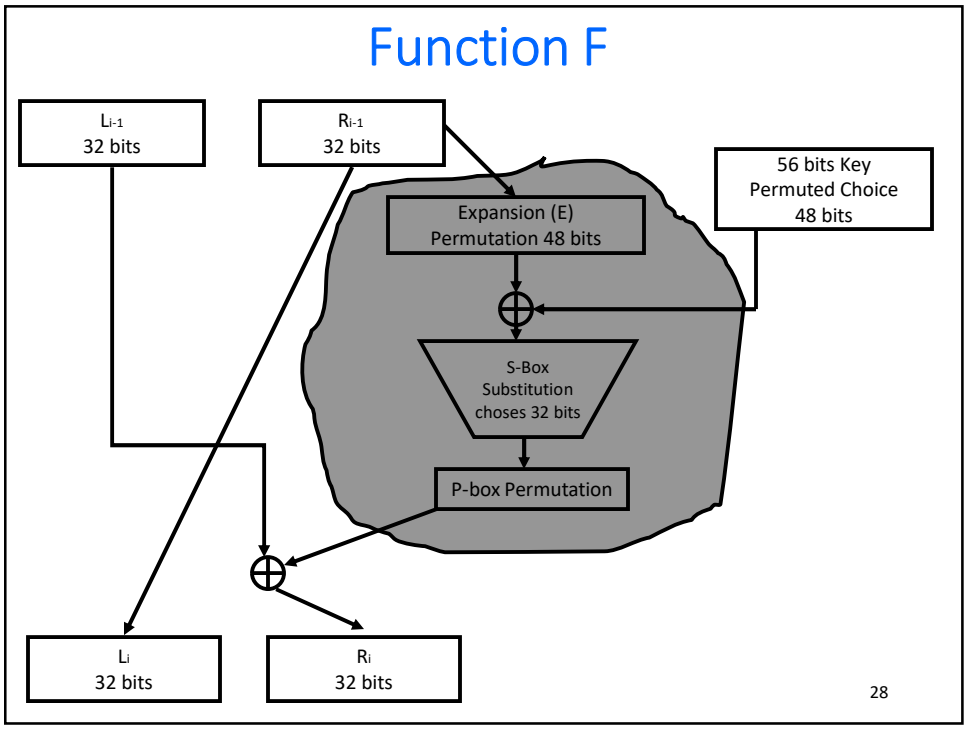


38

38

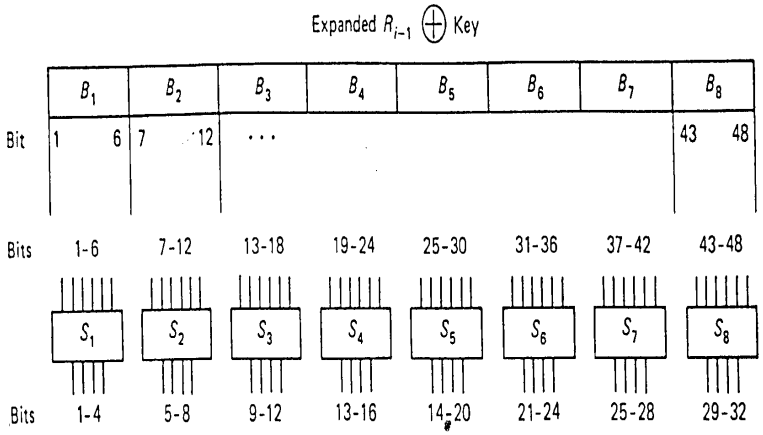

DES System
Encryption Process
64 Bit Plaintext
Initial Permutation
32 Bit L0
32 Bit R0
F(R0,K1)
32 Bit L1
32 Bit R1
32 Bit L15
32 Bit R15
F(R15,K16)
32 Bit L16
32 Bit R16
Final Permutation
64 Bit Ciphertext
Building Blocks
K1(48 bits)
K16(48 bits)
Key Schedule
64 Bit Key
Permutation Choice 1
56 Bit Key
28 Bit C0
28 Bit D0
Left Shift
Left Shift
C1
D1
Permuted Choice 2
C16
D16
Permuted Choice 2
27

39


Function F
Li-1
32 bits
Ri-1
32 bits
56 bits Key
Permuted Choice
48 bits
Expansion (E)
Permutation 48 bits
S-Box
Substitution
choses 32 bits
P-box Permutation
Li
32 bits
Ri
32 bits
28

40

DES Substitution Boxes Operation



29

41

Operation Tables of DES (IP, IP^{-1}, E and P)

Initial Pemutation (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

Inverse Initial Pemutation (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

Bit-Selection Table E

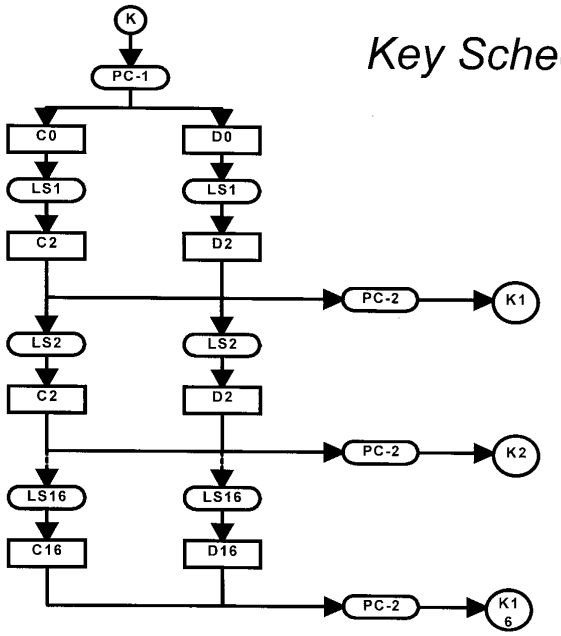
32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Permutation P

16	7	20	21
19	12	18	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

30

42



Key Schedule -- KS

31

43

Key schedule of shifts

Iteration(i)i	No. of shifts
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Key permutation PC-1

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

Key permutation PC-2

14	17	11	24	1	5
3	28	15	6	20	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	54
46	42	50	36	29	32

Operation Tables of DES
(Key Schedule,PC-1,PC-2)

32

44

Breaking DES (Cryptanalysis)

DES Key size = 56 bits

- Brute force = 2^{55} attempts on avg
- Differential cryptanalysis ➔ 2^{47} chosen plaintexts
- Linear cryptanalysis ➔ 2^{43} known plaintexts

•More than 16 rounds do not make it any stronger

•DES Key Problems:

- Weak keys (all 0s, all 1s, a few others)
- Key size = 56 bits = 8 * 7-bit ASCII

• Alphanumeric-only password converted to upper-case:

8 * ~5-bit chars = 40 bits

33

45

Recap: Average Time for Exhaustive Key Search (for Brute-Force Attacks)

Key Size (bits)	**Number of Alternative Keys**	**Time required at 10^6 Decr/µs**
32	$2^{32} = 4.3 \times 10^9$	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	10 hours
128	$2^{128} = 3.4 \times 10^{38}$	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	5.9×10^{30} years

Today, **> 80 bits** is considered "secure"

46

46