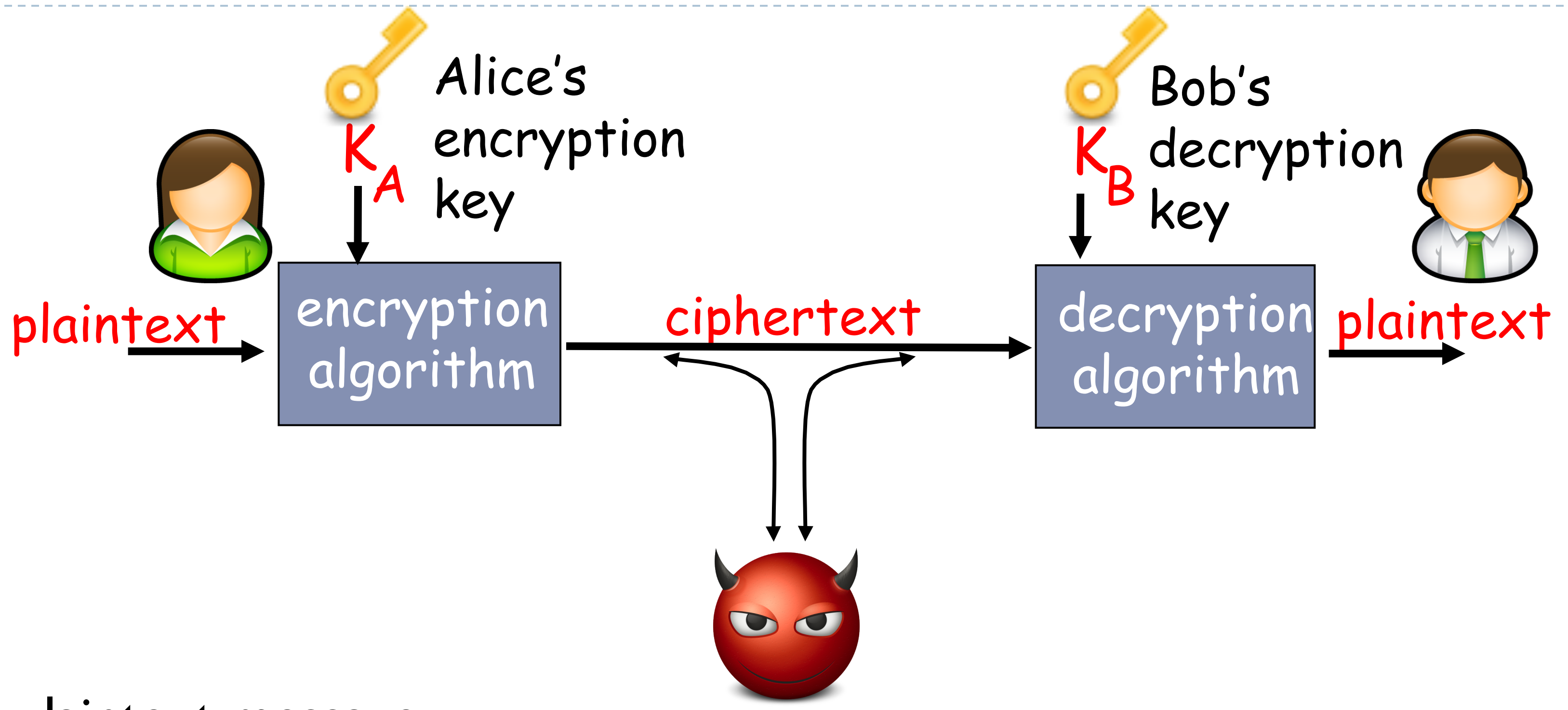
SYSTEM FAILURE
error 235553261...pending.....
fatal ER # 5444167QW32Z__WS @%$$

CSCI 4250/6250 – Fall 2013 Computer and Networks Security

INTRODUCTION TO CRYPTO
CHAPTER 8 (Goodrich)
CHAPTER 2-6 (Kaufman)
CHAPTER 8 (Kurose)

- Slides adapted from Kurose et al., Goodrich et al., and Kaufman et al.

The language of cryptography



m plaintext message

$K_A(m)$ ciphertext, encrypted with key K_A

$m = K_B(K_A(m))$

Basics

- Alternative Notation
  - Secret key K
  - Encryption function $E_K(P)$
  - Decryption function $D_K(C)$
  - Plaintext length typically the same as ciphertext length
  - Encryption and decryption are permutation functions (bijections) on the set of all n-bit arrays
- Efficiency
  - functions E_K and D_K should have efficient algorithms
- Consistency
  - Decrypting the ciphertext yields the plaintext
  - $D_K(E_K(P)) = P$

Simple encryption scheme (Ceasar cipher)

substitution cipher: substituting one thing for another

- monoalphabetic cipher: substitute one letter for another

```
plaintext:   abcdefghijklmnopqrstuvwxyz

ciphertext:  mnbvcxzasdfghjklpoiuytrewq
```

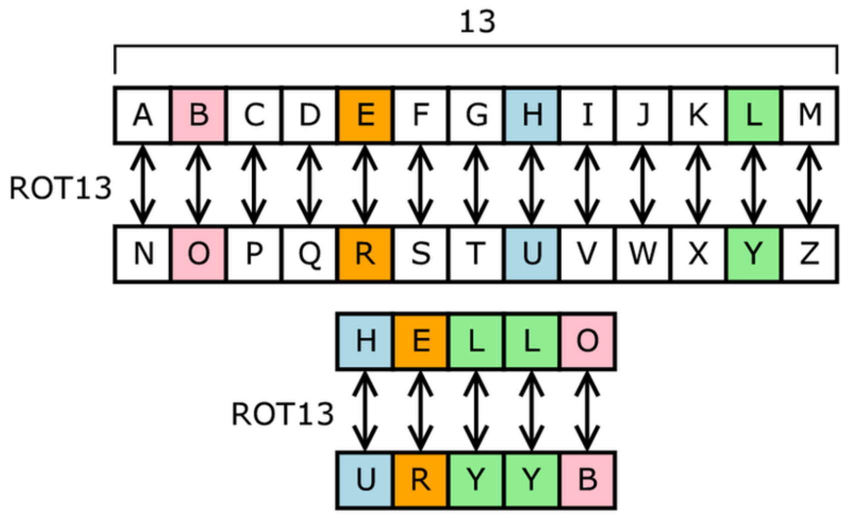
E.g.:

```
Plaintext: bob. i love you. alice
ciphertext: nkn. s gktc wky. mgsbc
```

Key: the mapping from the set of 26 letters to the set of 26 letters

Substitution Ciphers

- Each letter is uniquely replaced by another.
- ROT13 examaple:
  - CIAO → PVNB
- One popular substitution "cipher" for some Internet posts is ROT13.



Public domain image from http://en.wikipedia.org/wiki/File:ROT13.png

Polyalphabetic encryption

- n monoalphabetic cyphers, $M_1, M_2, \ldots, M_n$
- Cycling pattern:
  - e.g., n=4 $M_1,M_3,M_4,M_3,M_2; M_1,M_3,M_4,M_3,M_2;$
- For each new plaintext symbol, use subsequent monoalphabetic pattern in cyclic pattern
  - dog: d from M_1, o from M_3, g from M_4
- Key: the n ciphers and the cyclic pattern

- Example:
  - Vigenere cipher

Vigenere cipher

- Plaintext
  - ATTACKATDAWN
- Key
  - LEMON
- Keystream
  - LEMONLEMONLE…
- Ciphertext
  - LXFOPVEFRNHR

	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
A	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
B	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	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	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

Example from Wikipedia
http://en.wikipedia.org/wiki/Vigen%C3%A8re_cipher

Cryptography vs. Cryptanalysis

- Cryptographers invent new clever cryptographic schemes
  - Objective: make it infeasible to recover the plaintext
    - Computational difficulty: efficient to compute cipher-text, but hard to "reverse" without the key
- Cryptanalysis studies cryptographic schemes
  - Objective: try to find flaws in the schemes
    - E.g., recover some info about the plaintext, or recover the key
- **Fundamental Tenet of Cryptography**
  - "If lots of smart people have failed to solve a problem, then it probably won't be solved (soon)"

Breaking an encryption scheme

- Cipher-text only attack: Trudy has ciphertext that she can analyze
- Two approaches:
  - Search through all keys: must be able to differentiate resulting plaintext from gibberish
  - Statistical analysis

- Known-plaintext attack: trudy has some plaintext corresponding to some ciphertext
  - eg, in monoalphabetic cipher, trudy determines pairings for a,l,i,c,e,b,o,
- Chosen-plaintext attack: trudy can get the cyphertext for some chosen plaintext

The crypto algorithms is typically public. Only thing that is assumed to be secret is the key.

Attacks

- Attacker may have

Harder

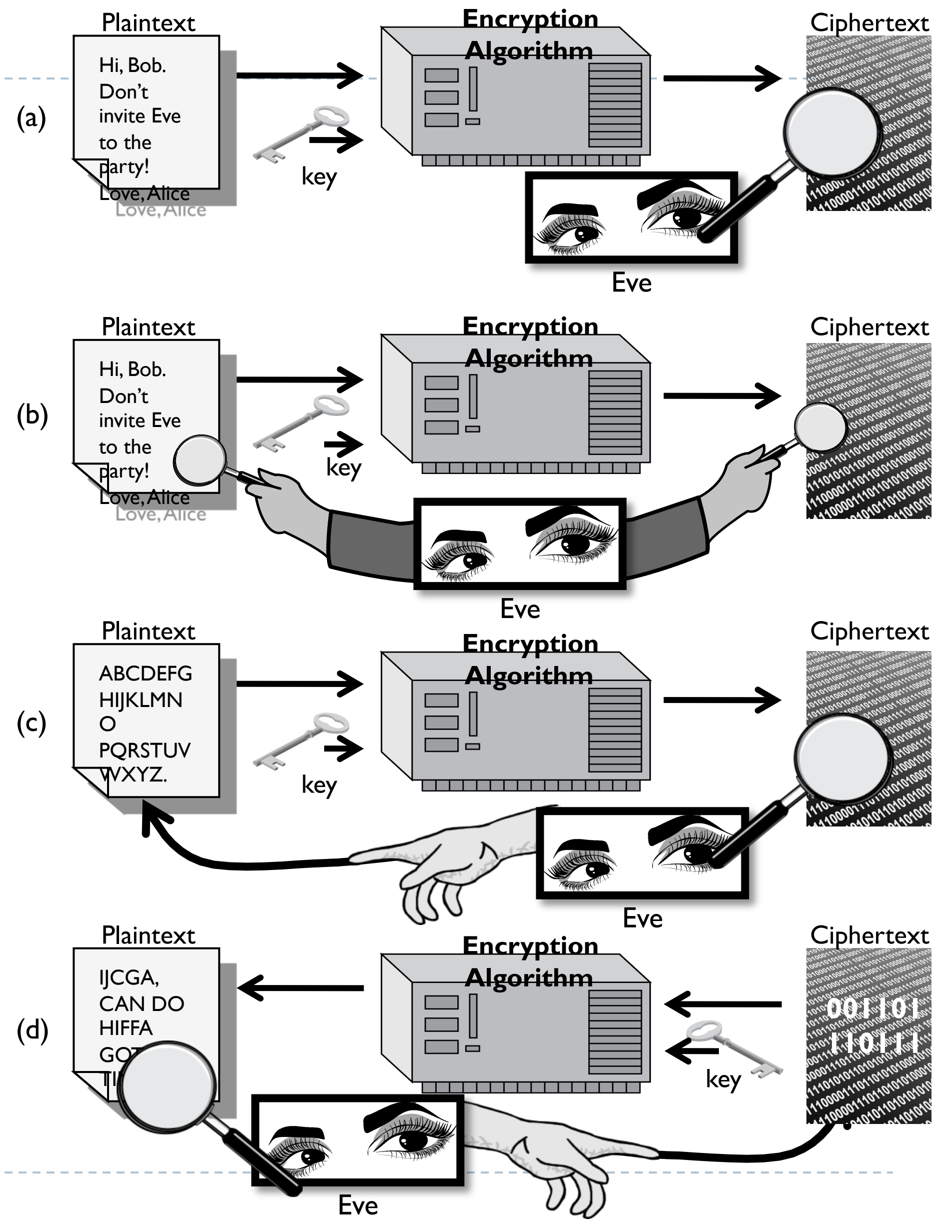
a) collection of ciphertexts (ciphertext only attack)

b) collection of plaintext/ ciphertext pairs (known plaintext attack)

c) collection of plaintext/ ciphertext pairs for plaintexts selected by the attacker (chosen plaintext attack)

d) collection of plaintext/ ciphertext pairs for ciphertexts selected by the attacker (chosen ciphertext attack)

Easier



Frequency Analysis

- Letters in a natural language, like English, are not uniformly distributed.
- Knowledge of letter frequencies, including pairs and triples can be used in cryptologic attacks against substitution ciphers.

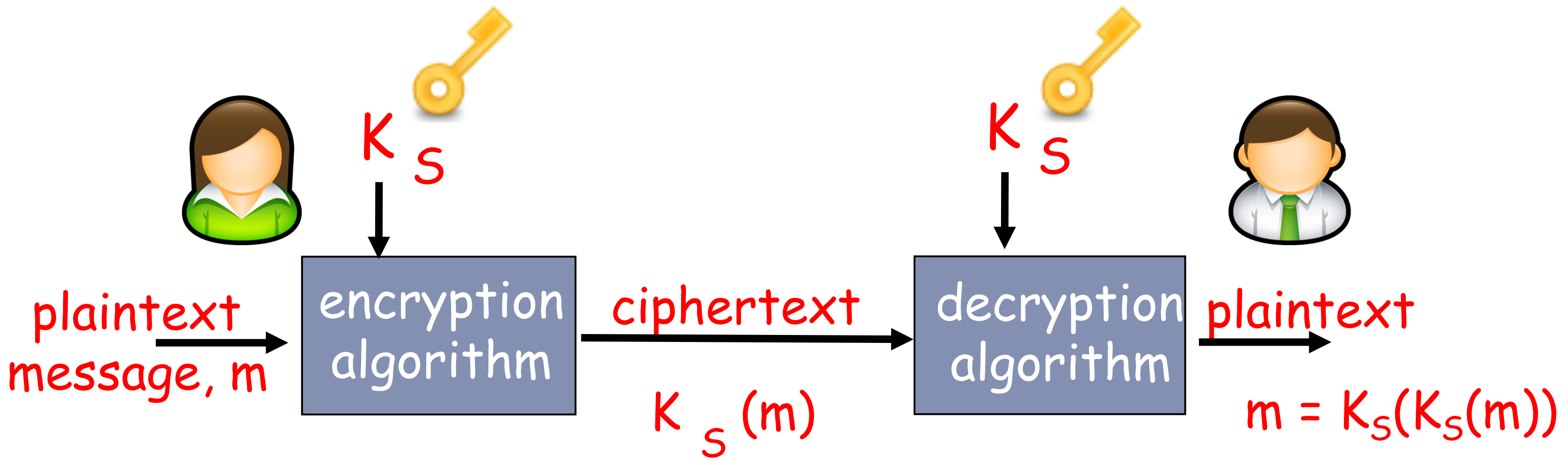
a:	8.05%	b:	1.67%	c:	2.23%	d:	5.10%
e:	12.22%	f:	2.14%	g:	2.30%	h:	6.62%
i:	6.28%	j:	0.19%	k:	0.95%	l:	4.08%
m:	2.33%	n:	6.95%	o:	7.63%	p:	1.66%
q:	0.06%	r:	5.29%	s:	6.02%	t:	9.67%
u:	2.92%	v:	0.82%	w:	2.60%	x:	0.11%
y:	2.04%	z:	0.06%				

8.1: Letter frequencies in the book *The Adventures of Tom Sawyer*, by Twain.

Types of Cryptography

- Crypto often uses keys:
  - Algorithm is known to everyone
  - Only "keys" are secret
- Public key cryptography
  - Involves the use of two keys
- Symmetric key cryptography
  - Involves the use of one key
- Hash functions
  - Involves the use of no keys
  - Nothing secret: How can this be useful?

Symmetric key cryptography



symmetric key crypto: Bob and Alice share same (symmetric) key: K_S

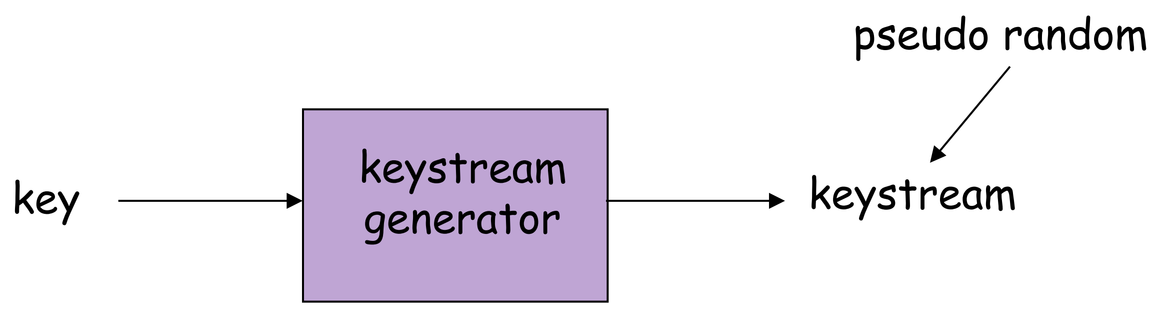
- e.g., key is knowing substitution pattern in mono alphabetic substitution cipher

Q: how do Bob and Alice agree on key value?

Two types of symmetric ciphers

- Stream ciphers
  - encrypt one bit at time
- Block ciphers
  - Break plaintext message in equal-size blocks
  - Encrypt each block as a unit

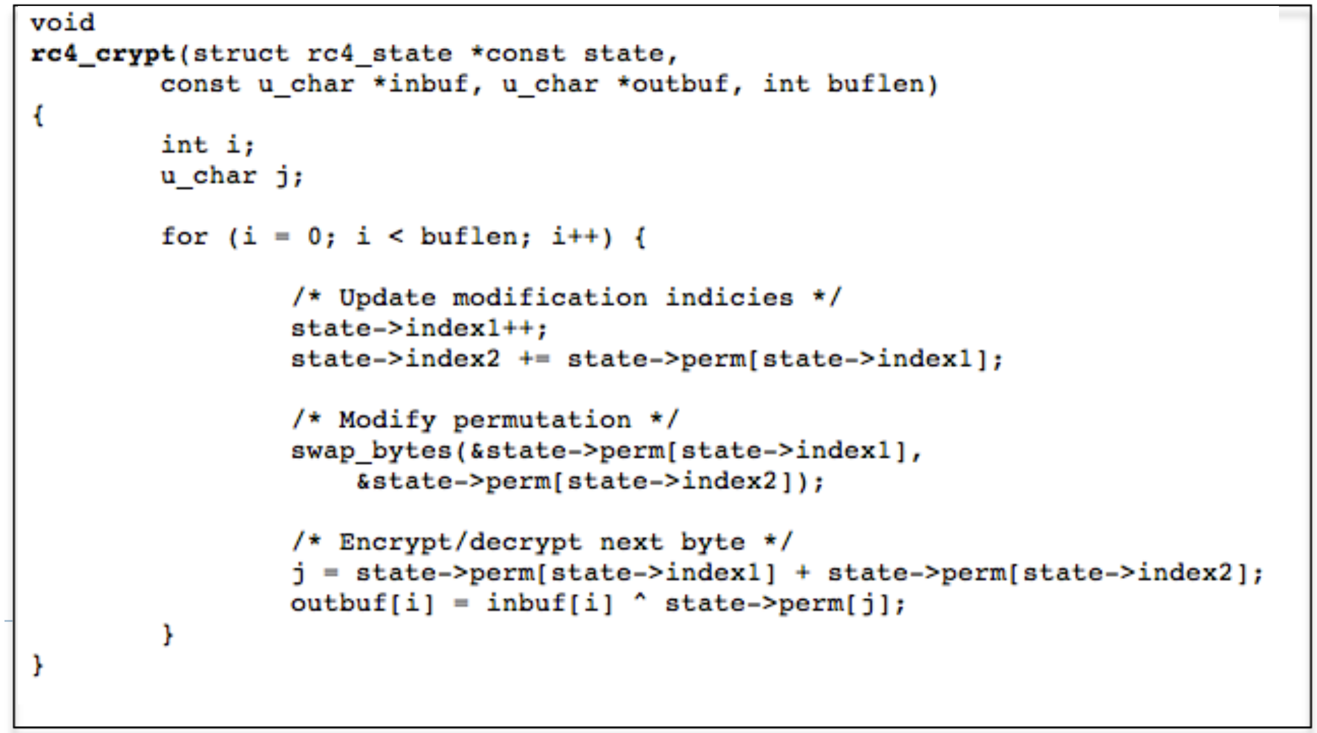
Stream Ciphers



- Combine each bit of keystream with bit of plaintext to get bit of ciphertext
  - m(i) = ith bit of message
  - ks(i) = ith bit of keystream
  - c(i) = ith bit of ciphertext
  - c(i) = ks(i) ⊕ m(i) (⊕ = exclusive or)
  - m(i) = ks(i) ⊕ c(i)
- Problem:
  - If attacker knows portion of plaintext P, she can replace it with desired malicious plaintext P’

RC4 Stream Cipher

- RC4 is a popular stream cipher
  - Extensively analyzed and considered good
  - Key can be from 1 to 256 bytes
  - Used in WEP for 802.11
  - Can be used in SSL



```
void
rc4_crypt(struct rc4_state *const state,
        const u_char *inbuf, u_char *outbuf, int buflen)
{
        int i;
        u_char j;

        for (i = 0; i < buflen; i++) {

                /* Update modification indicies */
                state->index1++;
                state->index2 += state->perm[state->index1];

                /* Modify permutation */
                swap_bytes(&state->perm[state->index1],
                    &state->perm[state->index2]);

                /* Encrypt/decrypt next byte */
                j = state->perm[state->index1] + state->perm[state->index2];
                outbuf[i] = inbuf[i] ^ state->perm[j];
        }
}
```

One-Time Pads

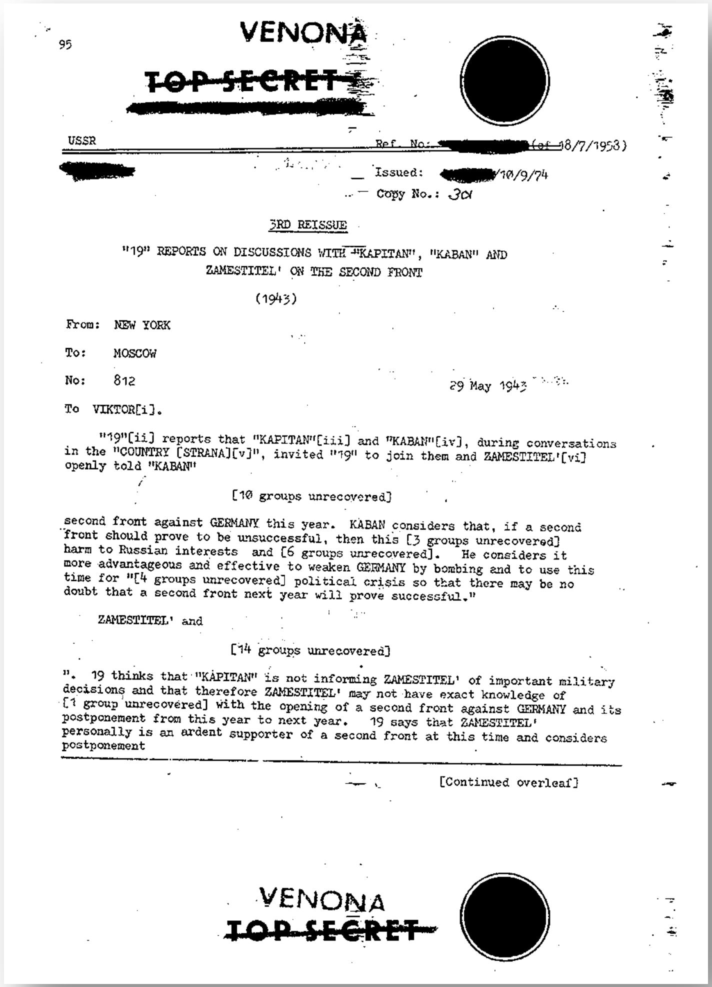
- There is one type of substitution cipher that is absolutely unbreakable.
  - The **one-time pad** was invented in 1917 by Joseph Mauborgne and Gilbert Vernam
  - We use a block of shift keys, $(k_1, k_2, \ldots, k_n)$, to encrypt a plaintext, M, of length n, with each shift key being chosen uniformly at random.
- Since each shift is random, every ciphertext is equally likely for any plaintext.

One-Time Pads

- Key is as long as the message to be sent
  - Stream of bits generated at random (not pseudo-random)
- Impossible to crack (perfect security?)
  - $H(M) = H(M|C)$
    - The ciphertext C provides no information about M
    - Given we only know C, every plaintext message is equally possible
  - Proven by Shannon
- Impractical
  - Keys need to be known to the receiver
  - Transferred through other means (e.g., paper)
  - Never reuse the same key

Weaknesses of the One-Time Pad

- In spite of their perfect security, one-time pads have some weaknesses
- The key has to be as long as the plaintext
- Keys can never be reused
  - Repeated use of one-time pads allowed the U.S. to break some of the communications of Soviet spies during the Cold War.



95

VENONA

~~TOP SECRET~~ [redacted] [redacted]

USSR Ref. No.: [redacted] (of 18/7/1953)

[redacted]

Issued: [redacted]/10/9/74

Copy No.: 3a

3RD REISSUE

"19" REPORTS ON DISCUSSIONS WITH "KAPITAN", "KABAN" AND ZAMESTITEL' ON THE SECOND FRONT

(1943)

From: NEW YORK

To: MOSCOW

No: 812 29 May 1943

To VIKTOR[i].

"19"[ii] reports that "KAPITAN"[iii] and "KABAN"[iv], during conversations in the "COUNTRY [STRANA][v]", invited "19" to join them and ZAMESTITEL'[vi] openly told "KABAN"

[10 groups unrecovered]

second front against GERMANY this year. KABAN considers that, if a second front should prove to be unsuccessful, then this [3 groups unrecovered] harm to Russian interests and [6 groups unrecovered]. He considers it more advantageous and effective to weaken GERMANY by bombing and to use this time for "[4 groups unrecovered] political crisis so that there may be no doubt that a second front next year will prove successful."

ZAMESTITEL' and

[14 groups unrecovered]

". 19 thinks that "KAPITAN" is not informing ZAMESTITEL' of important military decisions and that therefore ZAMESTITEL' may not have exact knowledge of [1 group unrecovered] with the opening of a second front against GERMANY and its postponement from this year to next year. 19 says that ZAMESTITEL' personally is an ardent supporter of a second front at this time and considers postponement

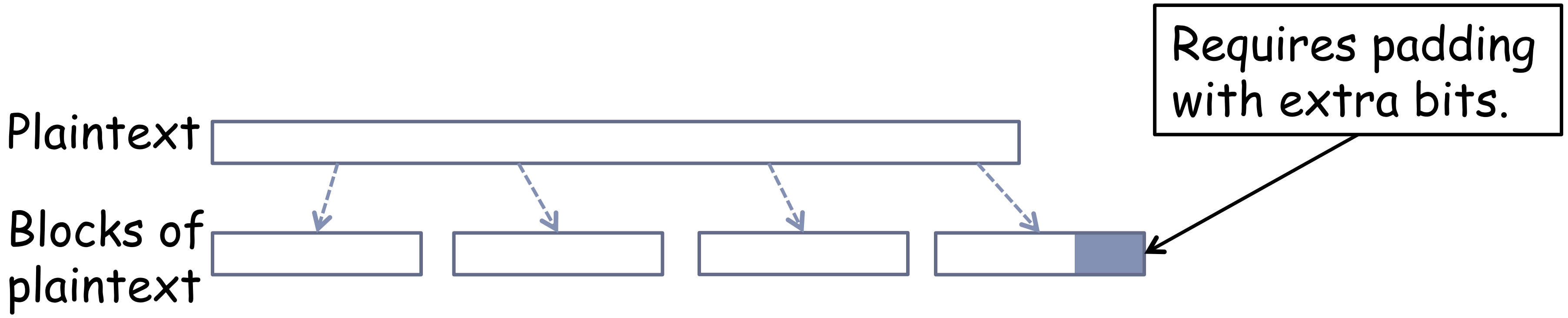
[Continued overleaf]

VENONA

~~TOP SECRET~~ [redacted]

Block Ciphers

- In a **block cipher:**
  - Plaintext and ciphertext have fixed length b (e.g., 128 bits)
  - A plaintext of length n is partitioned into a sequence of m **blocks**, P[0], …, P[m-1], where $n \le bm < n + b$
- Each message is divided into a sequence of blocks and encrypted or decrypted in terms of its blocks.



Padding

- Block ciphers require the length n of the plaintext to be a multiple of the block size b
- Padding the last block needs to be unambiguous (cannot just add zeroes)
- When the block size and plaintext length are a multiple of 8, a common padding method (PKCS5) is a sequence of identical bytes, each indicating the length (in bytes) of the padding
- Example for b = 128 (16 bytes)
  - Plaintext: "Roberto" (7 bytes)
  - Padded plaintext: "Roberto999999999" (16 bytes), where 9 denotes the number and not the character
- We need to always pad the last block, which may consist only of padding (http://tools.ietf.org/html/rfc2898)

Block ciphers

- Message to be encrypted is processed in blocks of k bits (e.g., 64-bit blocks).
- 1-to-1 mapping is used to map k-bit block of plaintext to k-bit block of ciphertext

Example with k=3:

input	output
000	110
001	111
010	101
011	100

input	output
100	011
101	010
110	000
111	001

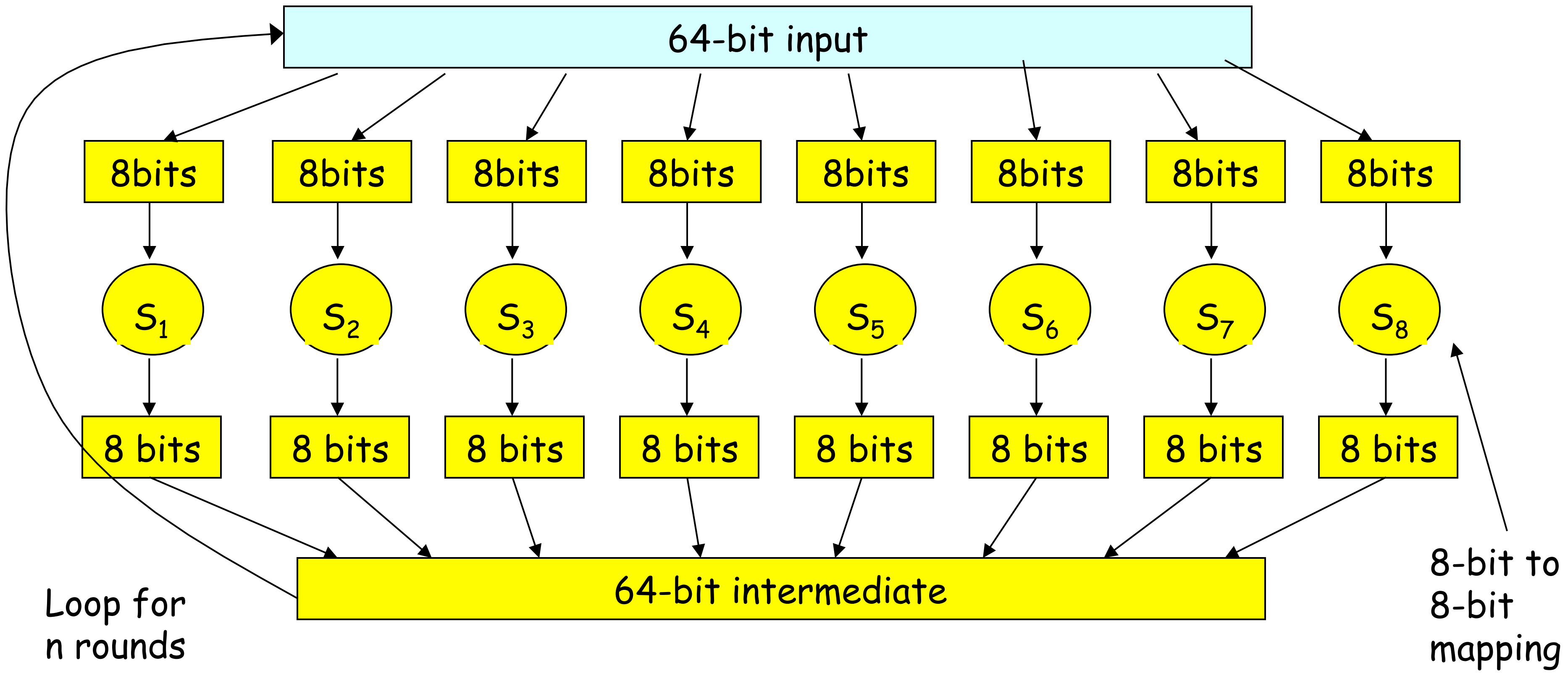
What is the ciphertext for 010110001111 ?

Block ciphers

- How many possible mappings are there for k=3?
  - How many 3-bit inputs?
  - How many permutations of the 3-bit inputs?
  - Answer: 40,320 ; not very many!
- In general, $2^k!$ mappings; huge for k=64
  - Hard to brute force!
- Storage Problem:
  - Table approach requires table with 2^{64} entries, each entry with 64 bits
  - It's like having a key that is 64×2^{64} bits long
- Table too big: instead use function that simulates a randomly permuted table

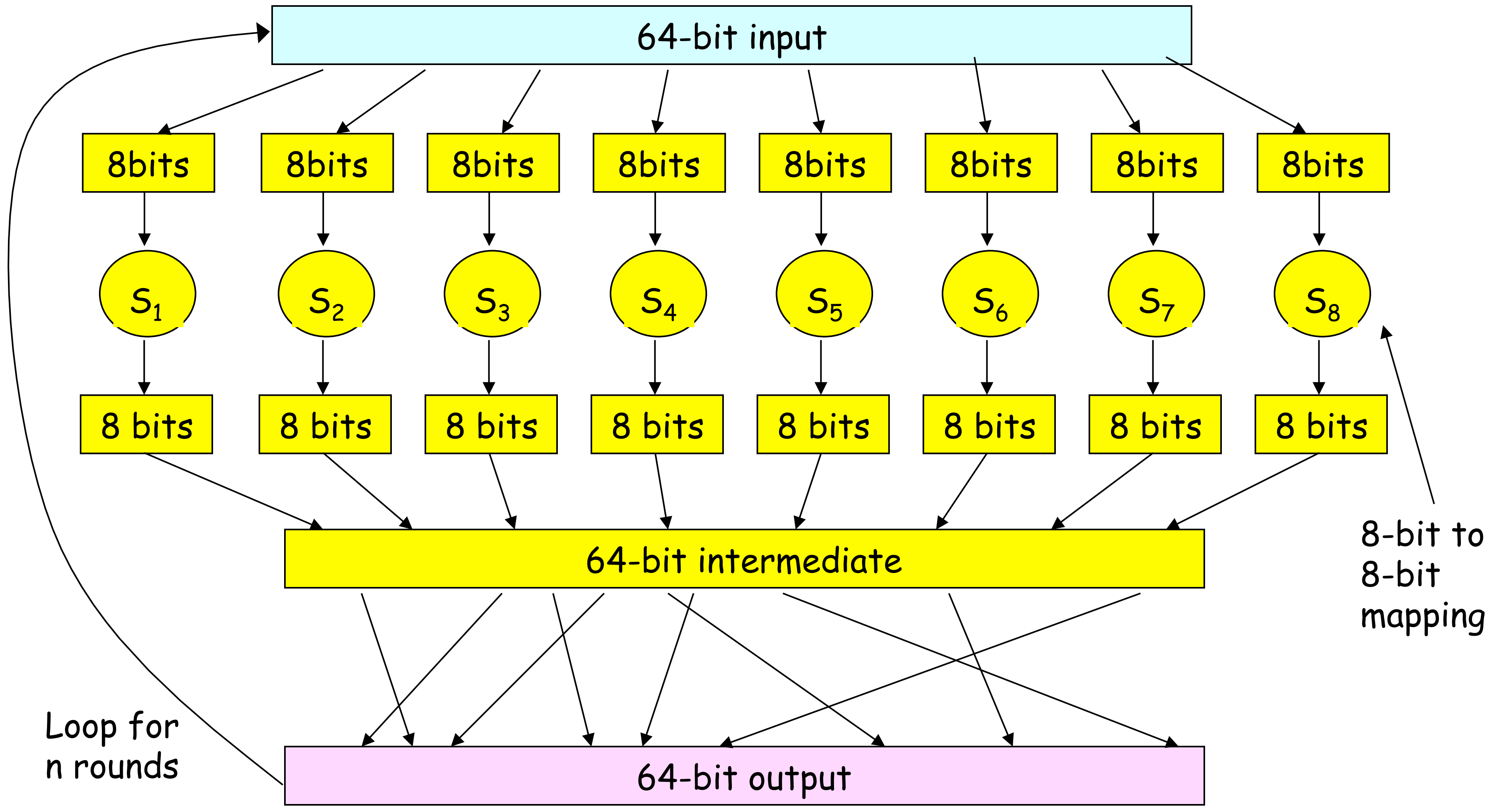
From Kaufman et al

Prototype function (Version 1)



From Kaufman et al

Prototype function (Version 2)



Why rounds?

- If only a single round, then one bit of input affects at most 8 bits of output.
- In 2nd round, the 8 affected bits get scattered (via permutation) and inputted into multiple substitution boxes.
- How many rounds?
  - How many times do you need to shuffle cards
  - Becomes less efficient as n increases

Symmetric key crypto: DES

DES: Data Encryption Standard

- US encryption standard [NIST 1993]
- 56-bit symmetric key (64 – 8 parity bits)
- 64-bit plaintext input blocks
- Can be used in a cipher block chaining (CBC) setting to encrypt longer messages

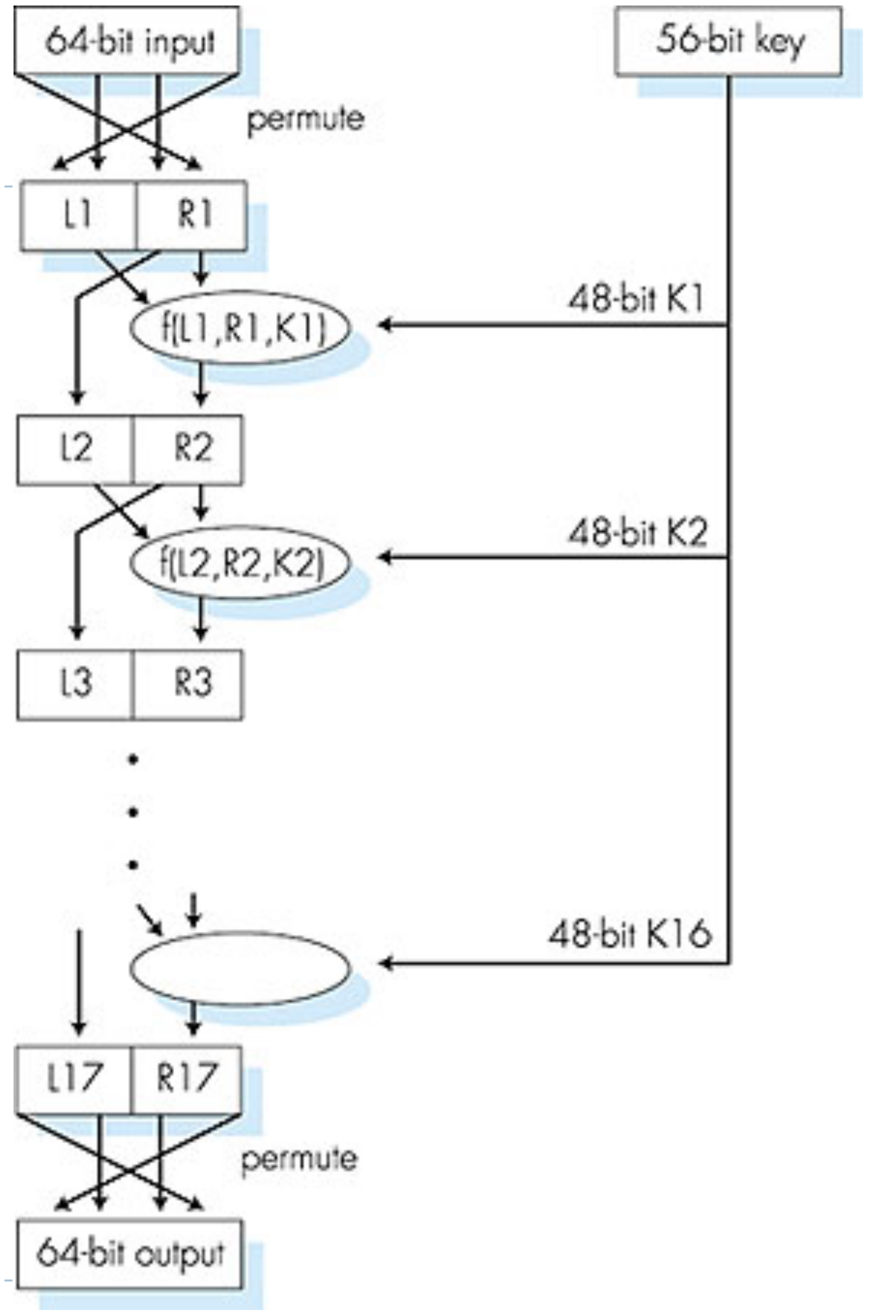
Symmetric key crypto: DES

DES operation

initial permutation

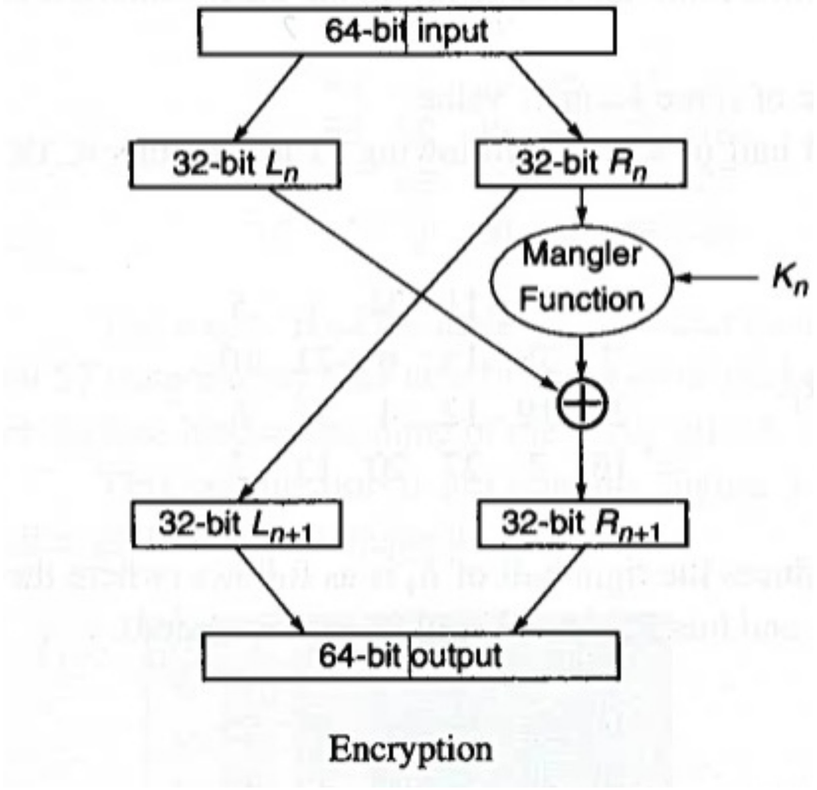
16 identical "rounds" of function application, each using different 48 bits of key

final permutation



DES Rounds

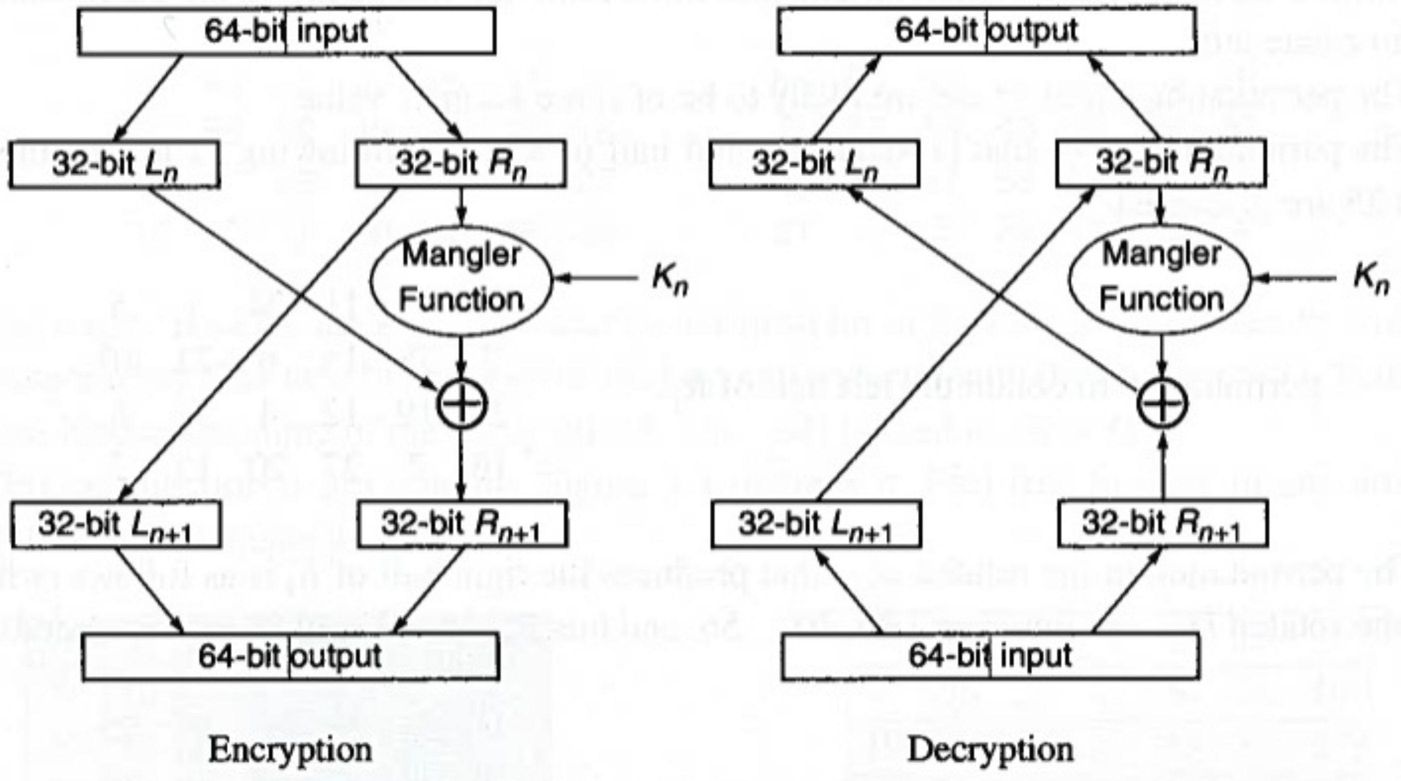
1-round Encryption and Decryption



See Kaufman et al. “Network Security, Private Communication in a Public World”

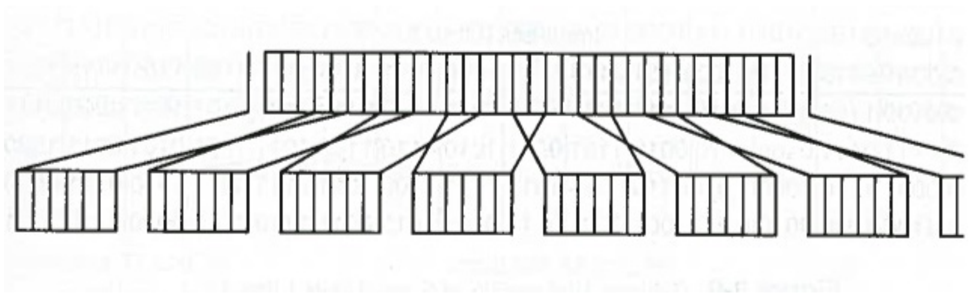
DES Rounds

1-round Encryption and Decryption

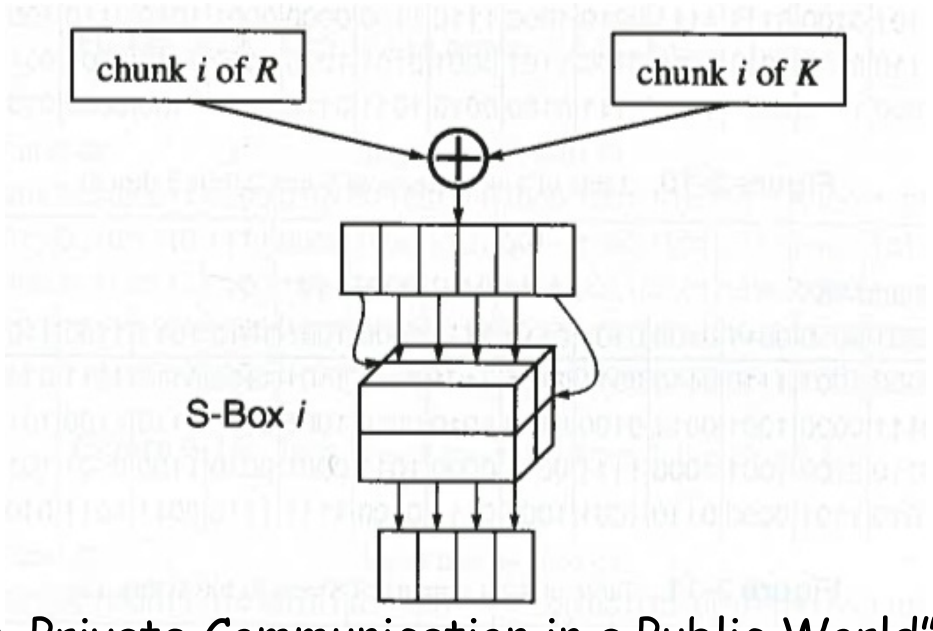


See Kaufman et al. "Network Security, Private Communication in a Public World"

DES Mangler Function



Expansion of R from 32 to 48 bits



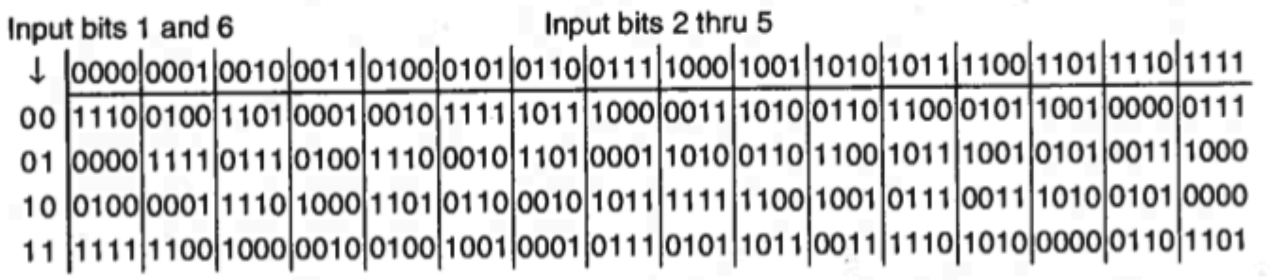
Expanded R and the Key are divided into eight 6-bit Chunks

Each 6-bit chunk is mapped into a 4-bit block

See Kaufman et al. “Network Security, Private Communication in a Public World”

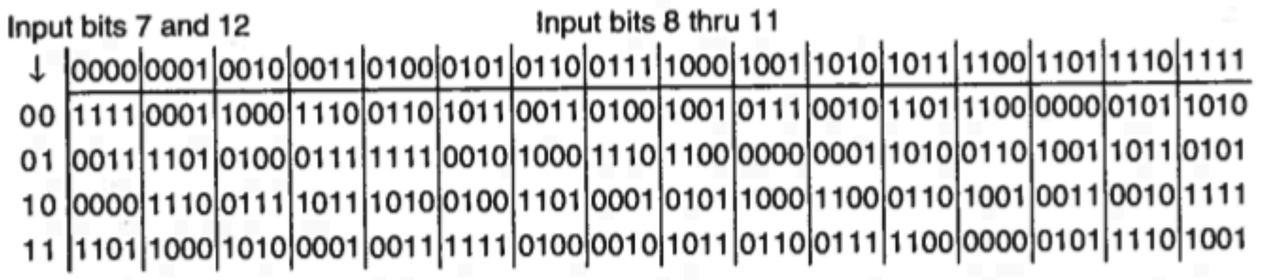
How does the S-box look like?

- There are 8 S-boxes (48/6)



Input bits 1 and 6	Input bits 2 thru 5															
↓	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

Figure 3-9. Table of 4-bit outputs of S-box 1 (bits 1 thru 4)



Input bits 7 and 12	Input bits 8 thru 11															
↓	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1111	0001	1000	1110	0110	1011	0011	0100	1001	0111	0010	1101	1100	0000	0101	1010
01	0011	1101	0100	0111	1111	0010	1000	1110	1100	0000	0001	1010	0110	1001	1011	0101
10	0000	1110	0111	1011	1010	0100	1101	0001	0101	1000	1100	0110	1001	0011	0010	1111
11	1101	1000	1010	0001	0011	1111	0100	0010	1011	0110	0111	1100	0000	0101	1110	1001

Figure 3-10. Table of 4-bit outputs of S-box 2 (bits 5 thru 8)

Generating Per-Round Keys

- Start with 56-bit key (64 - 8 parity bits)
  - Why 56 bits? Unknown…
- First divide 56-bit key into two 28-bit chunks
- Rotate bits for 16 rounds…
  - Some rounds rotate only by one bit, others rotate by two bits

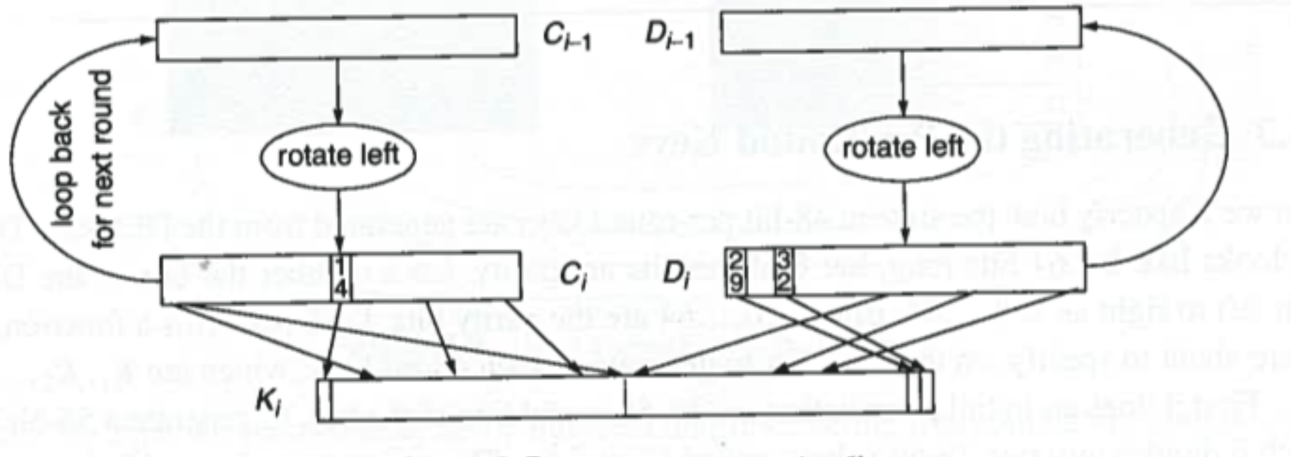
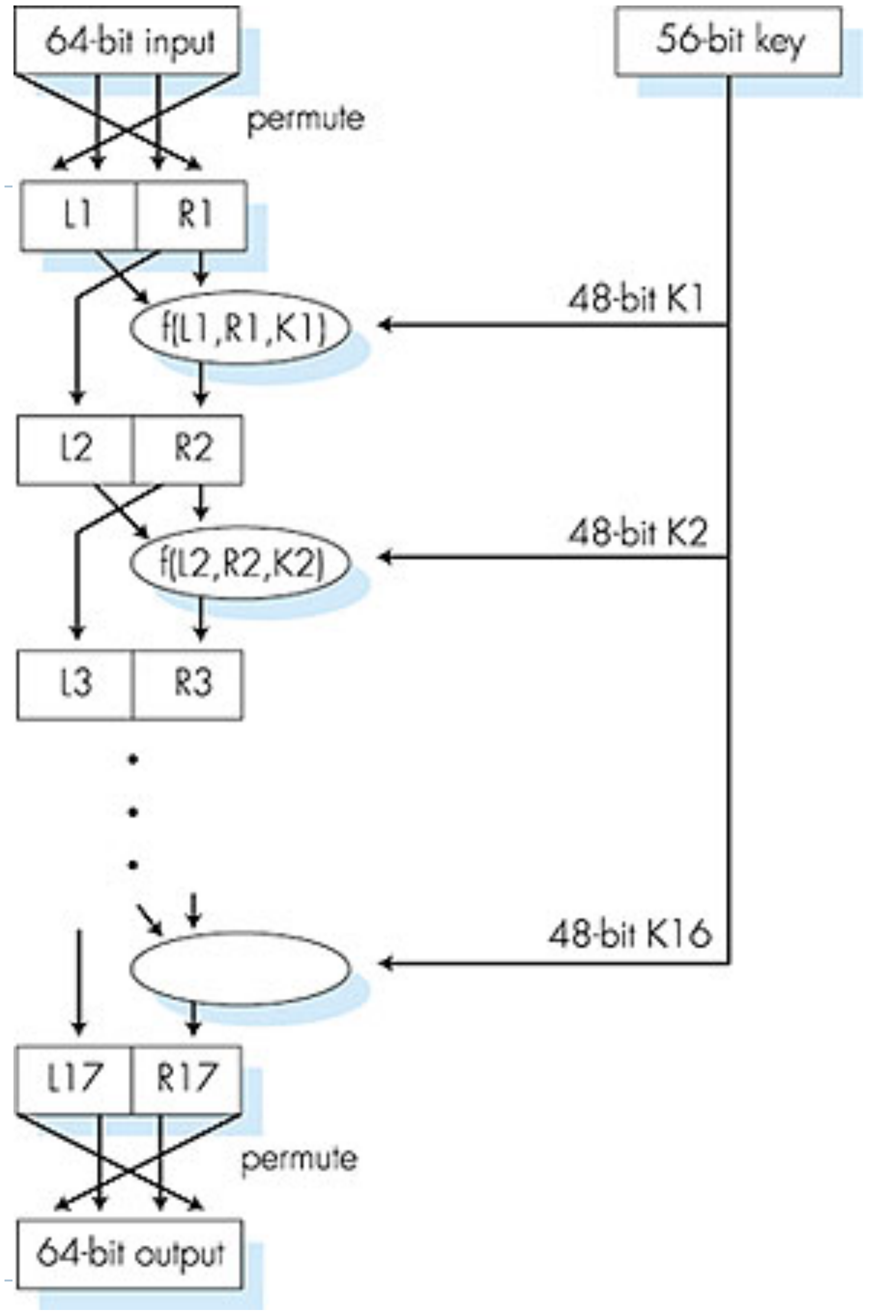


Figure 3-5. Round *i* for generating K_i

See Kaufman et al. "Network Security, Private Communication in a Public World"

Does DES work?



DES Security

- How secure is DES?
  - DES Challenge: 56-bit-key-encrypted phrase decrypted (brute force) in less than a day
  - No known good analytic attack
- making DES more secure:
  - 3DES: encrypt 3 times with 3 different keys (56*3=168 bits)

  (actually encrypt, decrypt, encrypt)
  - $c = K_c(K_b^{-1}(K_a(m)))$
  - $m = K_a^{-1}(K_b(K_c^{-1}(c)))$

3DES

- In practice only 2 keys are used
  - $c = K_a(K_b^{-1}(K_a(m)))$
  - $m = K_a^{-1}(K_b(K_a^{-1}(c)))$
  - It has been shown to be sufficiently secure
  - Avoids overhead of sending over 3 keys

- In DES we can *encrypt by decrypting* (???)
  - Using $c = K_a(K_b^{-1}(K_a(m)))$ allows for inter-operation with DES
  - Use Kb = Ka

- Why 3DES and not 120DES or 2DES?
  - 2DES has been proven not secure (takes only twice the time to brute-force a single-DES key)
  - 120DES would be very expensive from a computational point of view

The New York Times

NSA has some techniques in this area that we in the academic world do not. Certainly the fact that the NSA is pushing elliptic-curve cryptography is some indication that it can break them more easily.

WORLD U.S. N.Y. / REGION BUS

POLITICS EDUCATION TEXAS

N.S.A. Foils Much Internet Encryption

By NICOLE PERLROTH, JEFF LARSON and SCOTT SHANE
Published: September 5, 2013 | 471 Comments

The National Security Agency is winning its long-running secret war on encryption, using supercomputers, technical trickery, court orders and behind-the-scenes persuasion to undermine the major tools protecting the privacy of everyday communications in the Internet age, according to newly disclosed documents.

Enlarge This Image

Associated Press

This undated photo released by the United States government shows the National Security Agency campus in

The agency has circumvented or cracked much of the encryption, or digital scrambling, that guards global commerce and banking systems, protects sensitive data like trade secrets and medical records, and automatically secures the e-mails, Web searches, Internet chats and phone calls of Americans and others around the world, the documents show.

FACEB
TWITTE
GOOG
SAVE
E-MAIL
SHARE
PRINT
SINGLE
REPRIN

Beginning in 2000, as encryption tools were gradually blanketing the Web, the N.S.A. invested billions of dollars in a clandestine campaign to preserve its ability to eavesdrop. Having lost a public battle in the 1990s to insert its own "back door" in all encryption, it set out to accomplish the same goal by stealth.

The agency, according to the documents and interviews with industry officials, deployed custom-built, superfast computers to break codes, and began collaborating with technology companies in the United States and abroad to build entry points into their products. The documents do not identify which companies have participated.

Simultaneously, the N.S.A. has been deliberately weakening the international encryption standards adopted by developers. One goal in the agency's 2013 budget request was to "influence policies, standards and specifications for commercial public key technologies," the most common encryption method.

Cryptographers have long suspected that the agency planted vulnerabilities in a standard adopted in 2006 by the National Institute of Standards and Technology and later by the International Organization for Standardization, which has 163 countries as members.

"For the past decade, N.S.A. has led an aggressive, multipronged effort to break widely used Internet encryption technologies," said a 2010 memo describing a briefing about N.S.A. accomplishments for employees of its British counterpart, Government Communications Headquarters, or GCHQ. "Cryptanalytic capabilities are now coming online. Vast amounts of encrypted Internet data which have up till now been discarded are now exploitable."

Crypto modes

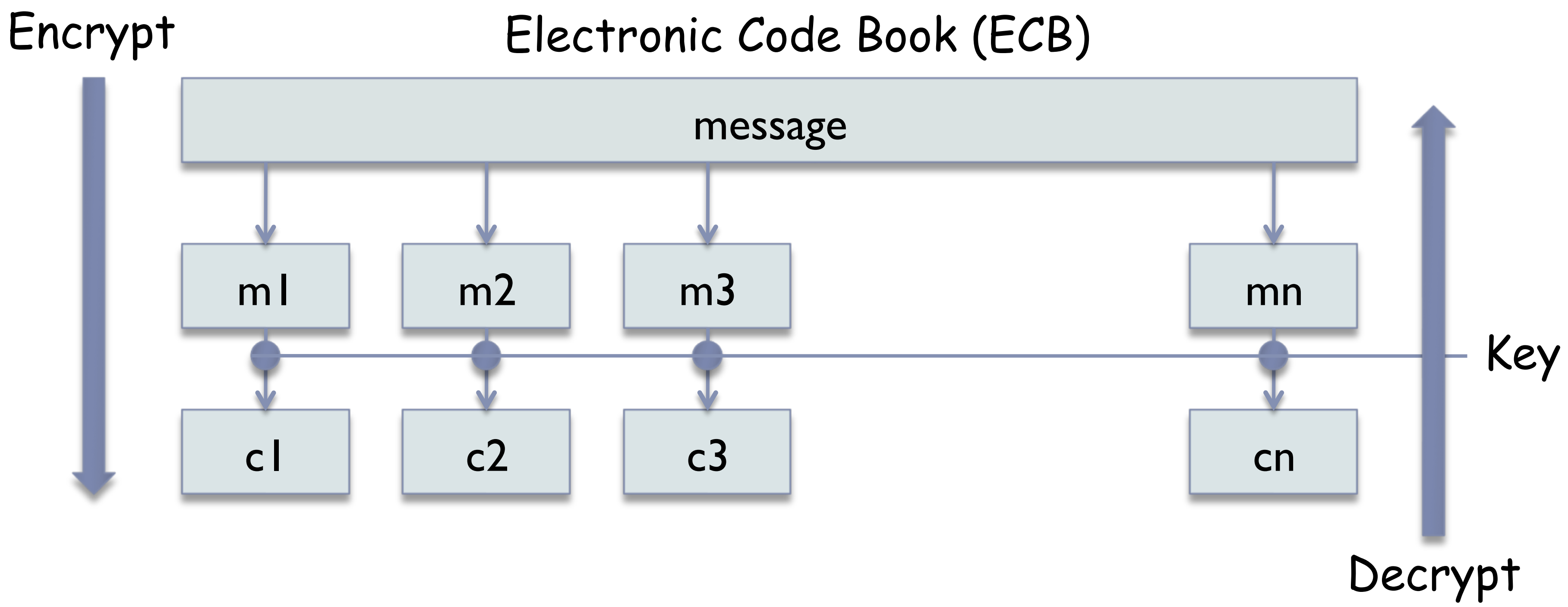
- Combining use of basic cipher for practical applications

- An application may need to
  - Be able to parallelize encryption and decryption
  - Preprocess as much as possible
  - Recover from bit errors/loss in the ciphertext
  - …

- Different modes provide different characteristics

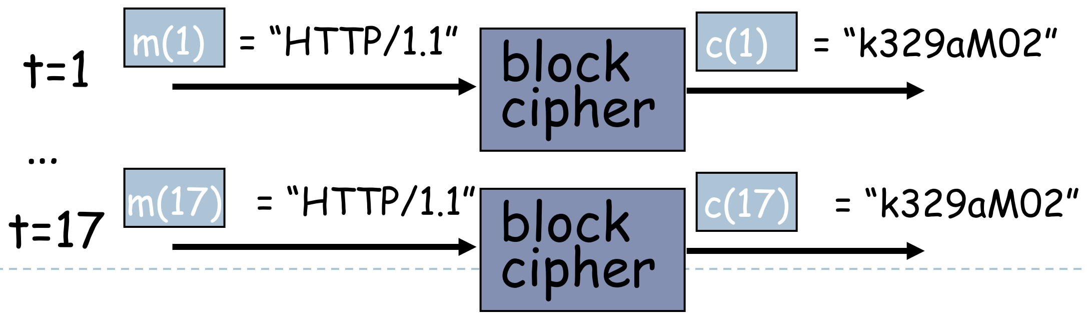
Encrypting a large message

- Why not just break message in 64-bit blocks, encrypt each block separately?



ECB

- Why not just break message in 64-bit blocks, encrypt each block separately?
  - The same plaintext always maps to the same ciphertext
    - in theory we can create a precomputed *code book* (one per key!)
  - Would be useful for random access files
    - ecryption and decryption trivially parallelizable
  - If same block of plaintext appears twice, will give same ciphertext
  - May facilitate cryptanalysis
  - we could swap things (e.g., swap salaries)



Strengths and Weaknesses of ECB

- Strengths:
  - Is very simple
  - Allows for parallel encryptions of the blocks of a plaintext
  - Can tolerate the loss or damage of a block

- Weakness:
  - Documents and images are not suitable for ECB encryption since patters in the plaintext are repeated in the ciphertext:

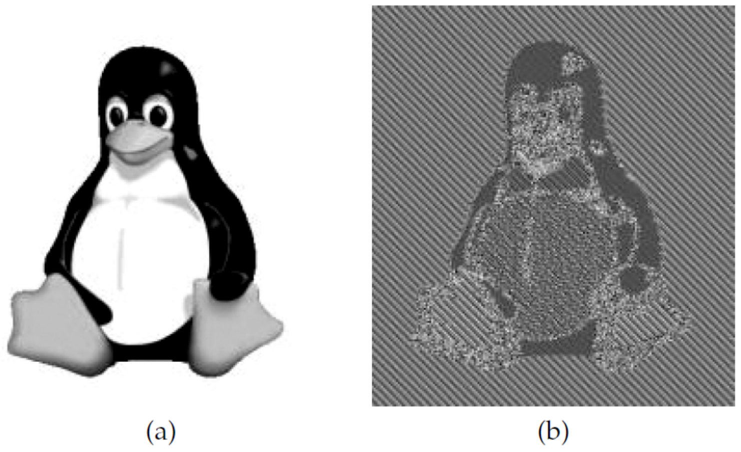


Figure 8.6: How ECB mode can leave identifiable patterns in a sequence of blocks: (a) An image of Tux the penguin, the Linux mascot. (b) An encryption of the Tux image using ECB mode. (The image in (a) is by Larry Ewing, lewing@isc.tamu.edu, using The Gimp; the image in (b) is by Dr. Juzam. Both are used with permission via attribution.)

Weaknesses of ECB

- Example: Assume attacker knows a block of plaintext and wants to modify replace it

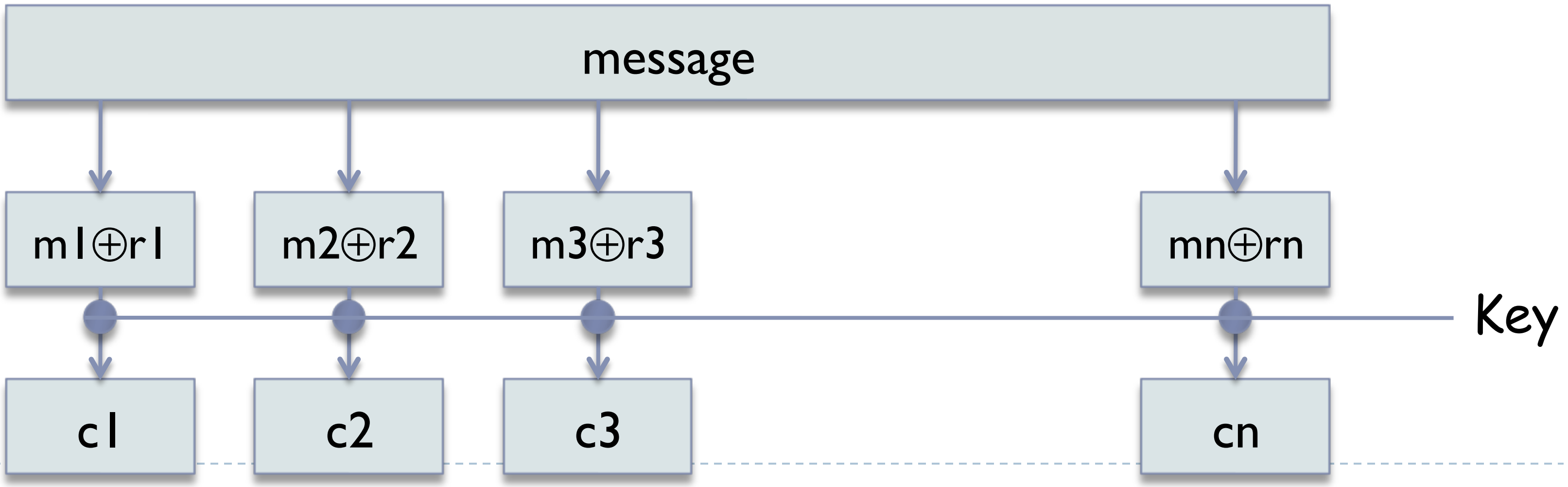
Jack Webb	$51,000	Jim Cook	$12,000
C1	C2	C3	C4

Jack Webb	$51,000	Jim Cook	$51,000
C1	C2	C3	C2

Encrypting a large message

- How about:
  - Generate random 64-bit number r(i) for each plaintext block m(i)
  - Calculate c(i) = K_S(m(i) ⊕ r(i))
  - Transmit c(i), r(i), i=1,2,…
  - At receiver: m(i) = K_S(c(i)) ⊕ r(i)
  - Problems:
    - inefficient, **need to send c(i) and r(i)**

Electronic Code Book (ECB)

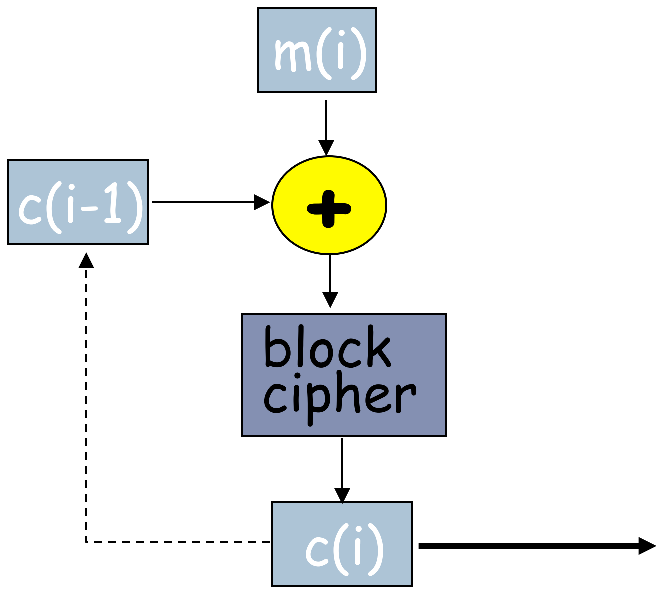


Cipher Block Chaining (CBC)

- CBC generates its own random numbers
  - Have encryption of current block depend on result of previous block
  - $c(i) = K_S(m(i) \oplus c(i-1))$
  - $m(i) = K_S(c(i)) \oplus c(i-1)$
- Forces same plaintext blocks to produce different ciphertext
- How do we encrypt first block?
  - Initialization vector (IV): random block = $c(0)$
  - IV does not have to be secret
- Change IV for each message (or session)
  - Guarantees that even if the same message is sent repeatedly, the ciphertext will be completely different each time

Cipher Block Chaining

- *cipher block chaining:* XOR ith input block, m(i), with previous block of cipher text, c(i-1)
  - c(0) transmitted to receiver in clear
  - what happens in "HTTP/1.1" scenario from above?



CBC

CBC Encryption

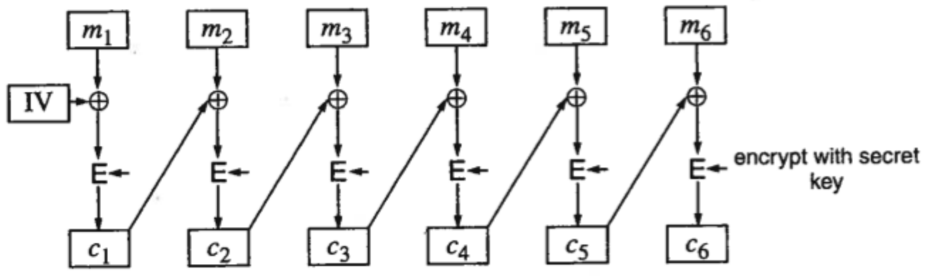


Figure 4-5. Cipher Block Chaining Encryption

See Kaufman et al. "Network Security, Private Communication in a Public World"

CBC

CBC Encryption

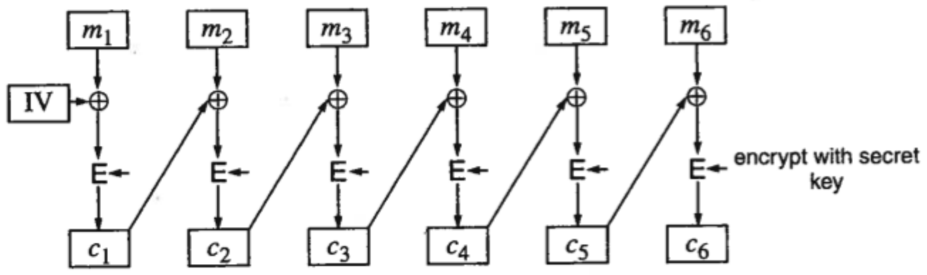


Figure 4-5. Cipher Block Chaining Encryption

CBC Decryption

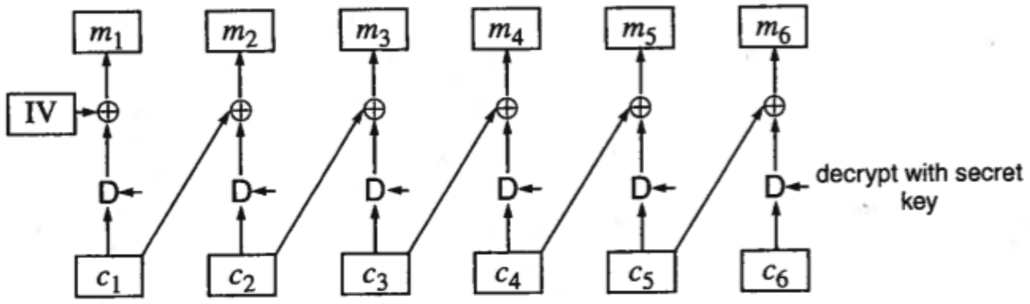


Figure 4-6. Cipher Block Chaining Decryption

See Kaufman et al. "Network Security, Private Communication in a Public World"

CBC: Threats

- CBC does not eliminate the possibility of somebody modifying the message *in transit*
- The attacker cannot swap blocks (e.g., to replace the IT guy's salary with the CEO salary), but can modify the ciphertext

- Example: Assume attacker knows a block of plaintext and wants to modify it

Jack Webb	IT Department	$51,000
Ci-1	Ci	Ci+1

- Changing Ci will modify Mi+1 in a predictable way
- However, Mi will be most likely garbled
  - The changed may be noticeable or not, the attacker may decide take his chances
- One possible defense
  - Attach one checksum block to the plaintext before encrypting
  - Changes in the plaintext will be detected with high probability

Strengths and Weaknesses of CBC

- Strengths:
  - Doesn't show patterns in the plaintext
  - Is the most common mode
  - Is fast and relatively simple
- Weaknesses:
  - CBC requires the reliable transmission of all the blocks sequentially
  - CBC is not suitable for applications that allow packet losses (e.g., music and video streaming)
  - Existence of Threats

Output Feedback Mode

- Use Block Cipher to generate key-stream (ks)
- $K(IV) = [b_0 \ldots b_n]$
- $K([b_0 \ldots b_n]) = b_{n+1} \ldots b_{2n}$
- etc.

- Keystream can be generated in advance, before message to be sent arrives
- Destination knows IV and K, therefore can generate same keystream

- Ciphertext generated as usual
  - Encryption: $c = m \oplus ks$
  - Decryption: $m = c \oplus ks$

- Potential problem
  - If somebody knows a portion P or the plaintext, that can be replaced with another "malicious" portion P'

Output Feedback Mode (k-bits)

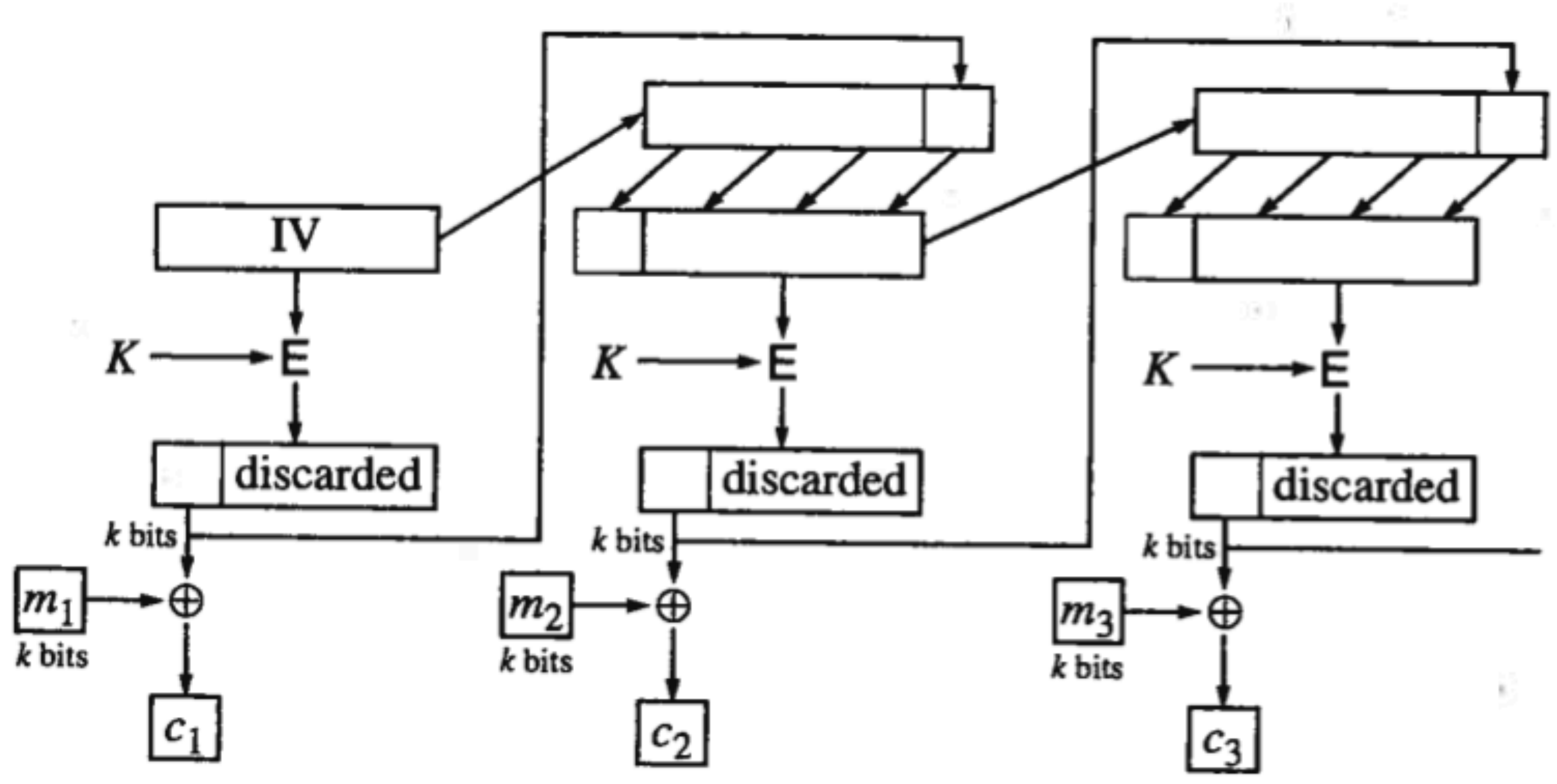


Figure 4-8. *k*-bit OFB

See Kaufman et al. "Network Security, Private Communication in a Public World"

Cipher Feedback Mode

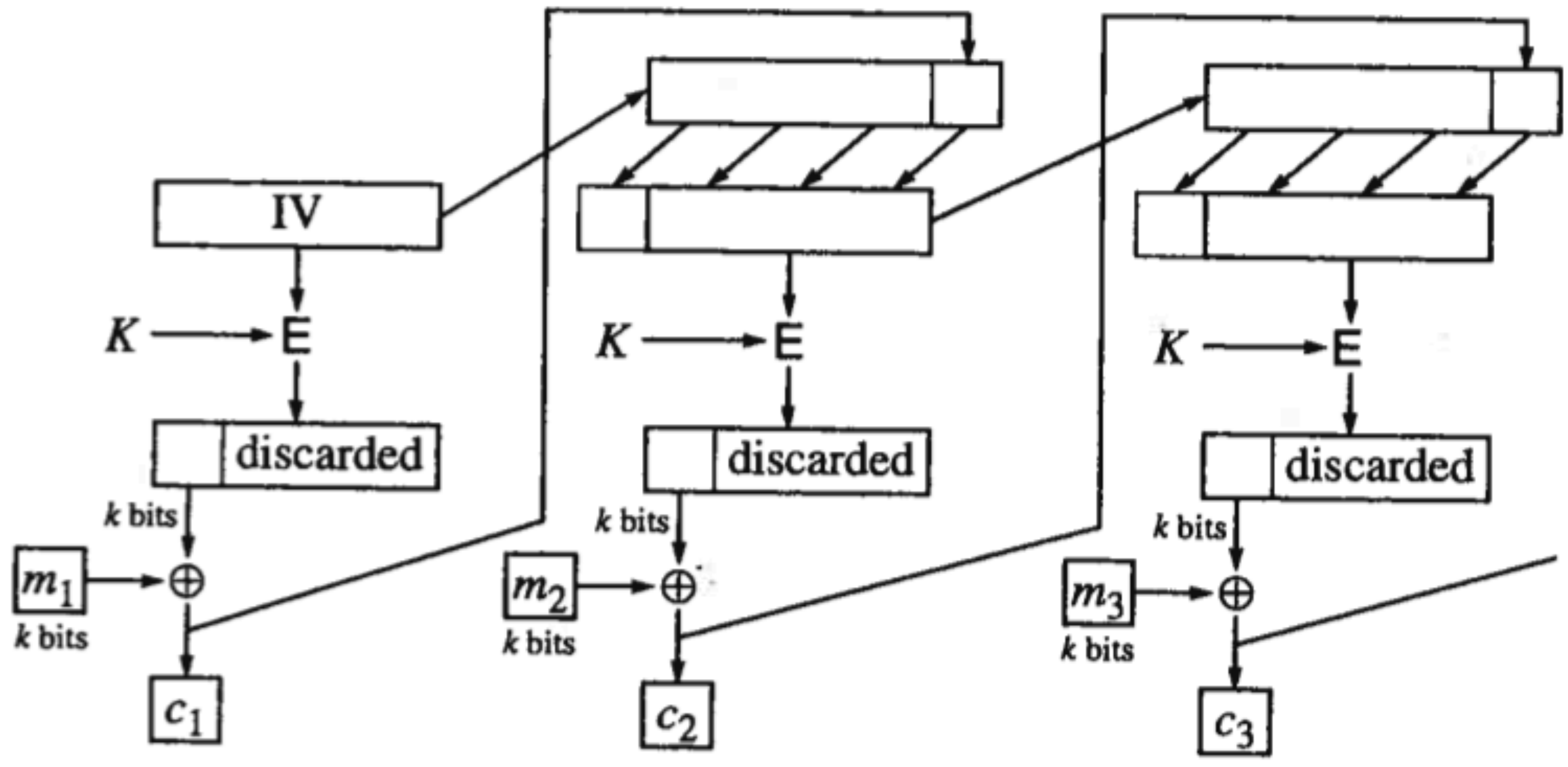


Figure 4-9. *k*-bit CFB

See Kaufman et al. “Network Security, Private Communication in a Public World”

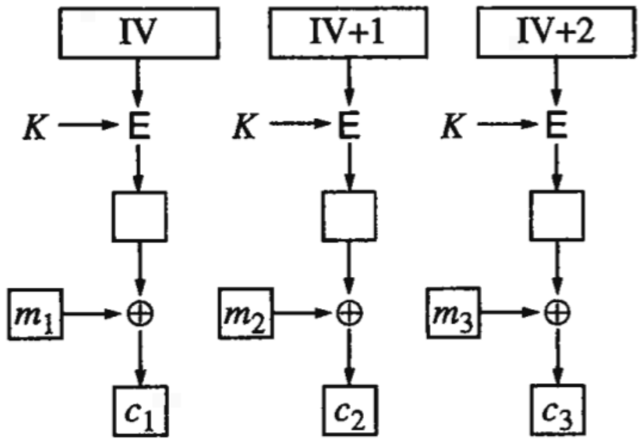
Cipher Feedback Mode (CFB)

- Keystream cannot be generated in advance
  - Need to wait for message to arrive

- Comparison with CBC and OFB
  - CBC/OFB: if bits of ciphertext lost in transmission, the entire rest of transmission is garbled
  - CFB: with 8-bit CFB, as long as the error is an integral number of bytes, things will re-sync. (1 bit error will affect 9 consecutive bytes)

Counter Mode (CTR)

- Similar to OFM
- Encrypts increments of IV to generate keystream
- Advantages:
  - Decryption can start anywhere, as long as you know the block number you are considering
  - Useful in case of encrypted random access files, for example



See Kaufman et al. “Network Security, Private Communication in a Public World”

Summary

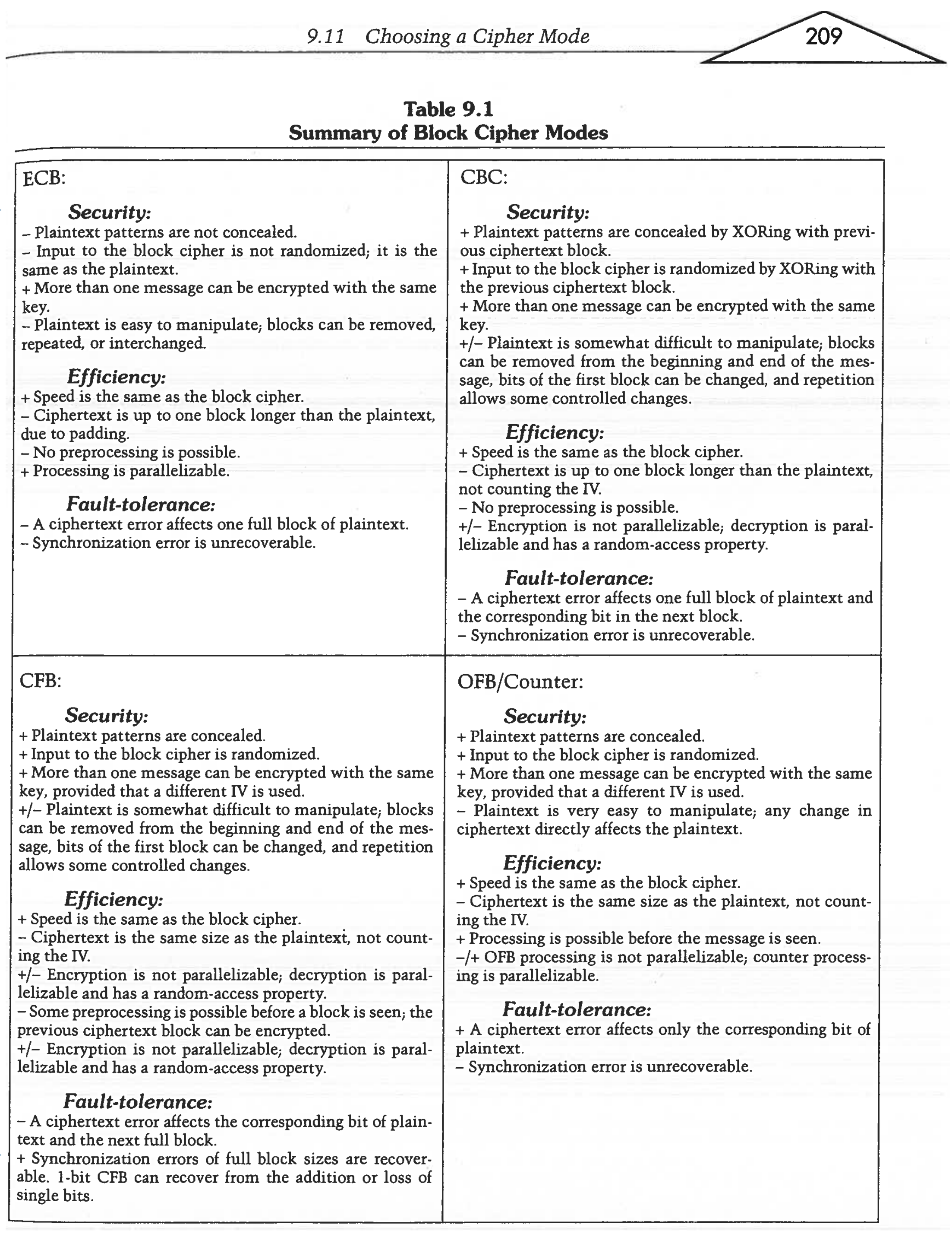


Table 9.1
Summary of Block Cipher Modes

ECB: ***Security:*** – Plaintext patterns are not concealed. – Input to the block cipher is not randomized; it is the same as the plaintext. + More than one message can be encrypted with the same key. – Plaintext is easy to manipulate; blocks can be removed, repeated, or interchanged. ***Efficiency:*** + Speed is the same as the block cipher. – Ciphertext is up to one block longer than the plaintext, due to padding. – No preprocessing is possible. + Processing is parallelizable. ***Fault-tolerance:*** – A ciphertext error affects one full block of plaintext. – Synchronization error is unrecoverable.	CBC: ***Security:*** + Plaintext patterns are concealed by XORing with previous ciphertext block. + Input to the block cipher is randomized by XORing with the previous ciphertext block. + More than one message can be encrypted with the same key. +/– Plaintext is somewhat difficult to manipulate; blocks can be removed from the beginning and end of the message, bits of the first block can be changed, and repetition allows some controlled changes. ***Efficiency:*** + Speed is the same as the block cipher. – Ciphertext is up to one block longer than the plaintext, not counting the IV. – No preprocessing is possible. +/– Encryption is not parallelizable; decryption is parallelizable and has a random-access property. ***Fault-tolerance:*** – A ciphertext error affects one full block of plaintext and the corresponding bit in the next block. – Synchronization error is unrecoverable.
CFB: ***Security:*** + Plaintext patterns are concealed. + Input to the block cipher is randomized. + More than one message can be encrypted with the same key, provided that a different IV is used. +/– Plaintext is somewhat difficult to manipulate; blocks can be removed from the beginning and end of the message, bits of the first block can be changed, and repetition allows some controlled changes. ***Efficiency:*** + Speed is the same as the block cipher. – Ciphertext is the same size as the plaintext, not counting the IV. +/– Encryption is not parallelizable; decryption is parallelizable and has a random-access property. – Some preprocessing is possible before a block is seen; the previous ciphertext block can be encrypted. +/– Encryption is not parallelizable; decryption is parallelizable and has a random-access property. ***Fault-tolerance:*** – A ciphertext error affects the corresponding bit of plaintext and the next full block. + Synchronization errors of full block sizes are recoverable. 1-bit CFB can recover from the addition or loss of single bits.	OFB/Counter: ***Security:*** + Plaintext patterns are concealed. + Input to the block cipher is randomized. + More than one message can be encrypted with the same key, provided that a different IV is used. – Plaintext is very easy to manipulate; any change in ciphertext directly affects the plaintext. ***Efficiency:*** + Speed is the same as the block cipher. – Ciphertext is the same size as the plaintext, not counting the IV. + Processing is possible before the message is seen. –/+ OFB processing is not parallelizable; counter processing is parallelizable. ***Fault-tolerance:*** + A ciphertext error affects only the corresponding bit of plaintext. – Synchronization error is unrecoverable.

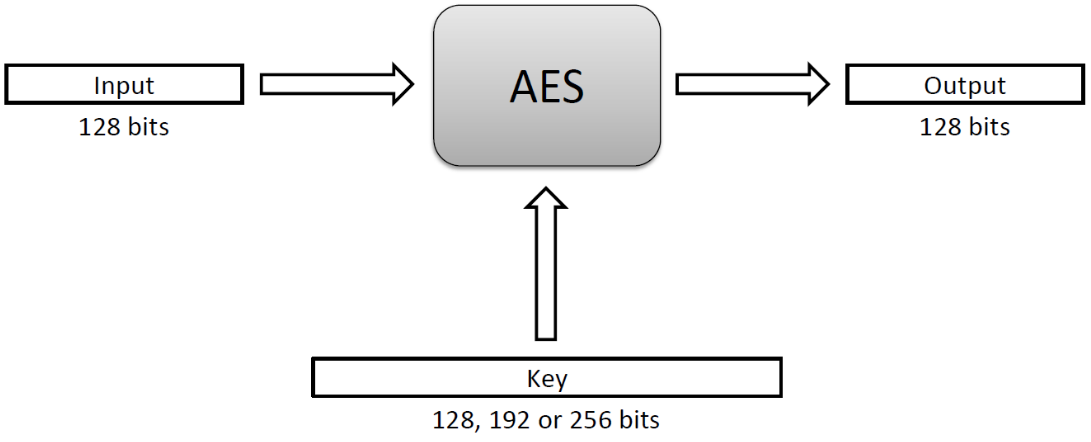
From "Applied Cryptography", 2nd edition
Bruce Schneier
Wiley

AES: Advanced Encryption Standard

- In 1997, the U.S. National Institute for Standards and Technology (NIST) put out a public call for a replacement to DES.
- It narrowed down the list of submissions to five finalists, and ultimately chose an algorithm that is now known as the **Advanced Encryption Standard** (**AES**).
- new (Nov. 2001) symmetric-key NIST standard, replacing DES
  - **Nice mathematical justification for design choices**
- processes data in 128 bit blocks
- 128, 192, or 256 bit keys
- brute force decryption (try each key) taking 1 sec on DES, takes 149 trillion years for AES

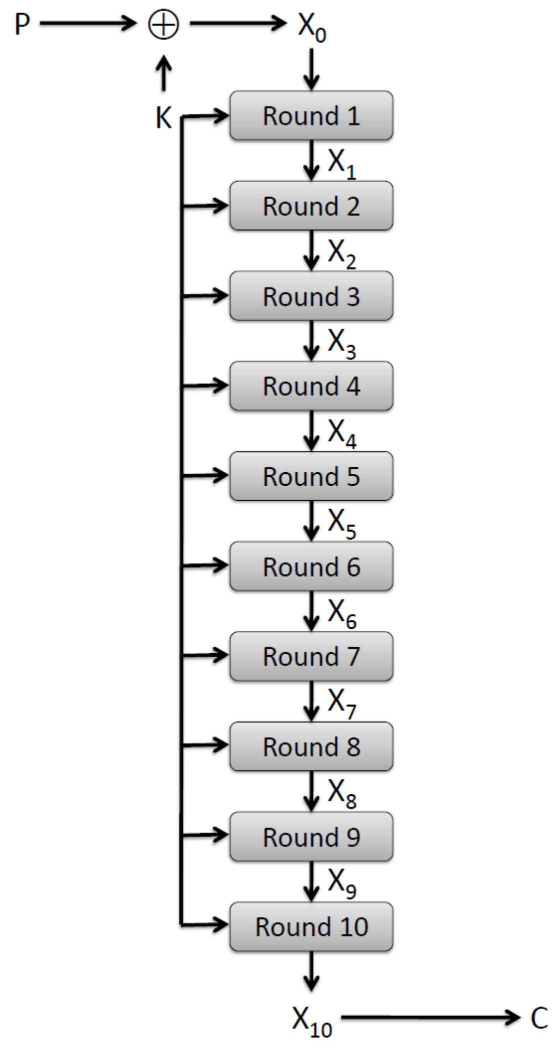
The Advanced Encryption Standard (AES)

- AES is a block cipher that operates on 128-bit blocks. It is designed to be used with keys that are 128, 192, or 256 bits long, yielding ciphers known as AES-128, AES-192, and AES-256.



AES Round Structure

- The 128-bit version of the AES encryption algorithm proceeds in ten rounds.
- Each round performs an invertible transformation on a 128-bit array, called **state**.
- The initial state X_0 is the XOR of the plaintext P with the key K:
- $X_0 = P \text{ XOR } K$.
- Round i (i = 1, …, 10) receives state X_{i-1} as input and produces state X_i.
- The ciphertext C is the output of the final round: $C = X_{10}$.



AES Rounds

- Each round is built from four basic steps:

1. **SubBytes step**: an S-box substitution step
2. **ShiftRows step**: a permutation step
3. **MixColumns step**: a matrix multiplication step
4. **AddRoundKey step**: an XOR step with a **round key** derived from the 128-bit encryption key

Key Exchange

- Enable Alice to communicate with Bob using shared key
  - The key cannot be transmitted in clear
  - It must be either encrypted when transmitted, or derived in a way that a third party cannot derive the same key
  - Alice and Bob may rely on a trusted third party, e.g., Cathy
  - The cryptosystem and protocols are publicly known

- First Attempt to Key Exchange
  - Alice and Cathy share a secret Ka
  - Cathy and Bob share a secret Kb

1. Alice >> Cathy : Ka(request for session key to Bob)
2. Cathy >> Alice : Ka(Ks) | Kb(Ks)
3. Alice >> Bob : Kb(Ks)
4. Alice can now privately send message M to Bob using Ks
   1. Alice >> Bob : Ks(M)

See Bishop "Introduction to Computer Security"

Key Exchange

- Problem: Replay Attack
  - Eve records (3) and Ks(M), which was sent by Alice to Bob
  - Eve >> Bob: Kb(Ks)
  - Eve >> Bob: Ks(M)
  - If M = “Deposit $500k in Roberto’s account”, we have a problem!

- Needham-Schroeder protocol
  1. Alice >> Cathy : “Alice” | “Bob” | Rand1
  2. Cathy >> Alice : Ka(“Alice” | “Bob” | Rand1 | Ks | Kb(“Alice” | Ks))
  3. Alice >> Bob : Kb(“Alice” | Ks)
  4. Bob >> Alice : Ks(Rand2)
  5. Alice >> Bob : Ks(Rand2-1)

See Bishop “Introduction to Computer Security”