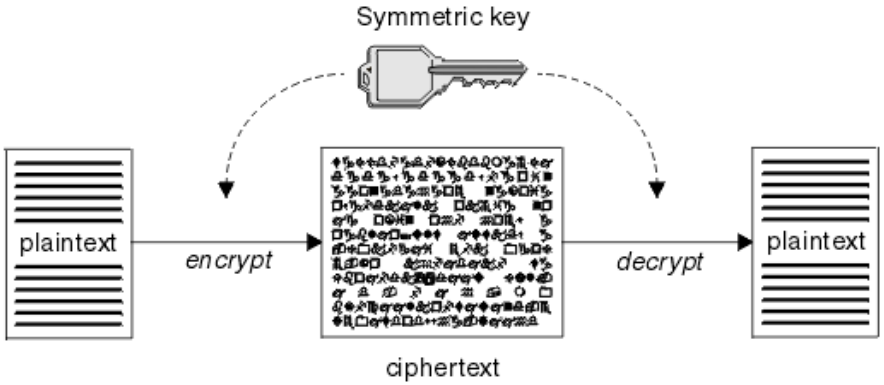


Computer Security

Symmetric Encryption

If you reveal your secrets to the wind, you should not blame the wind for revealing them to the trees.
—Kahlil Gibran

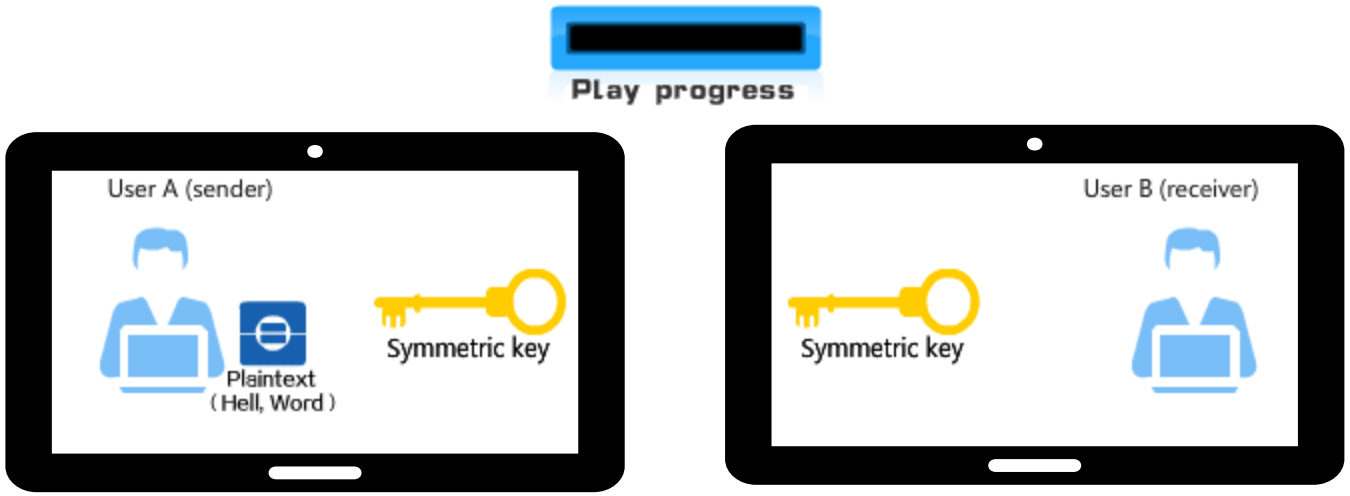
Tamer ABUHMED
Department of Computer Science & Engineering
Sungkyunkwan University

SUNG KYUN KWAN UNIVERSITY

Outline

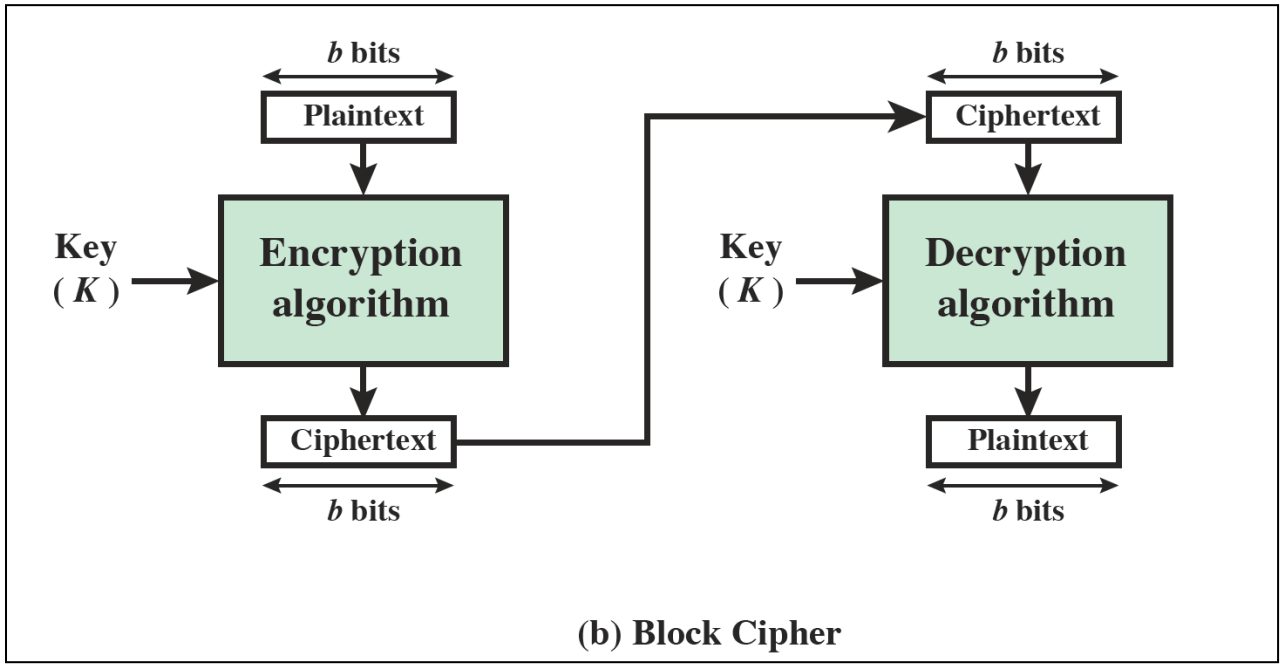
- Introduction to modern symmetric cryptosystems
- DES (Data Encryption Standard) cryptosystem
- 3DES (Triple DES)
- Blowfish
- RC family
- IDEA
- AES

Symmetric encryption



Modern Ciphers

- Block ciphers vs. Stream Ciphers
- Block ciphers operate on a block of data
  - entire block must be available before processing

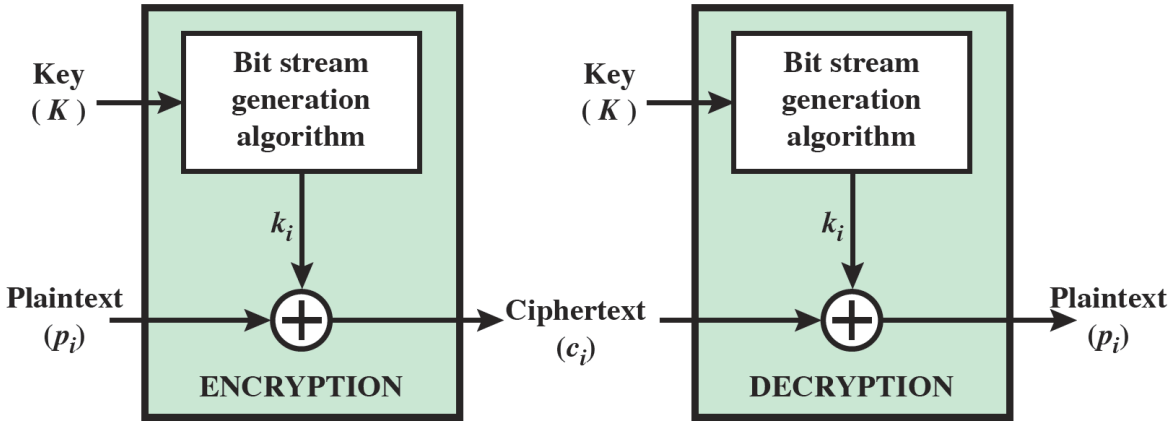


(b) Block Cipher

Encrypt/ Decrypt tool

Modern Ciphers

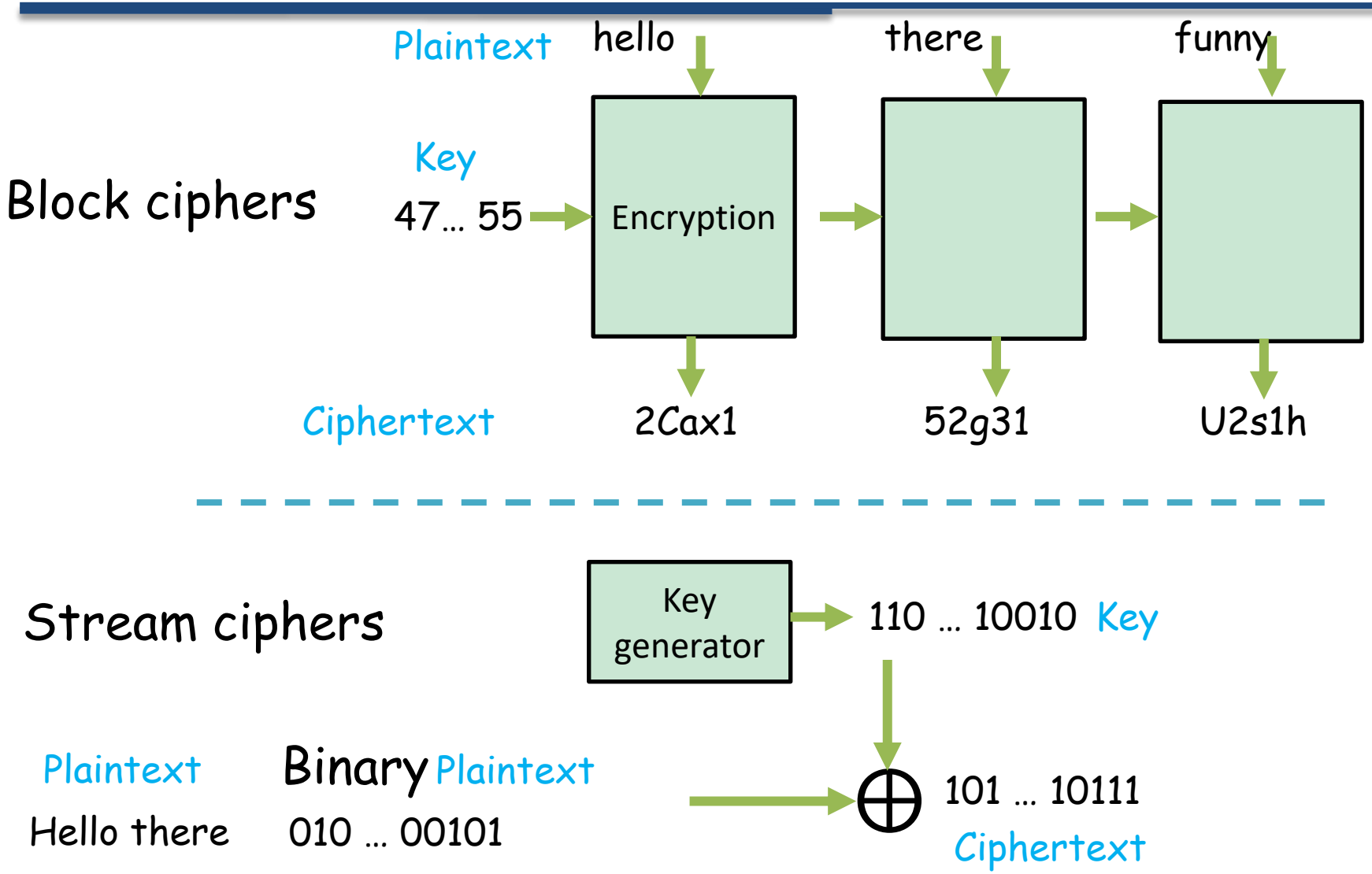
- Stream ciphers process messages one bit or byte at a time when en/decrypting
  - need not wait the entire block



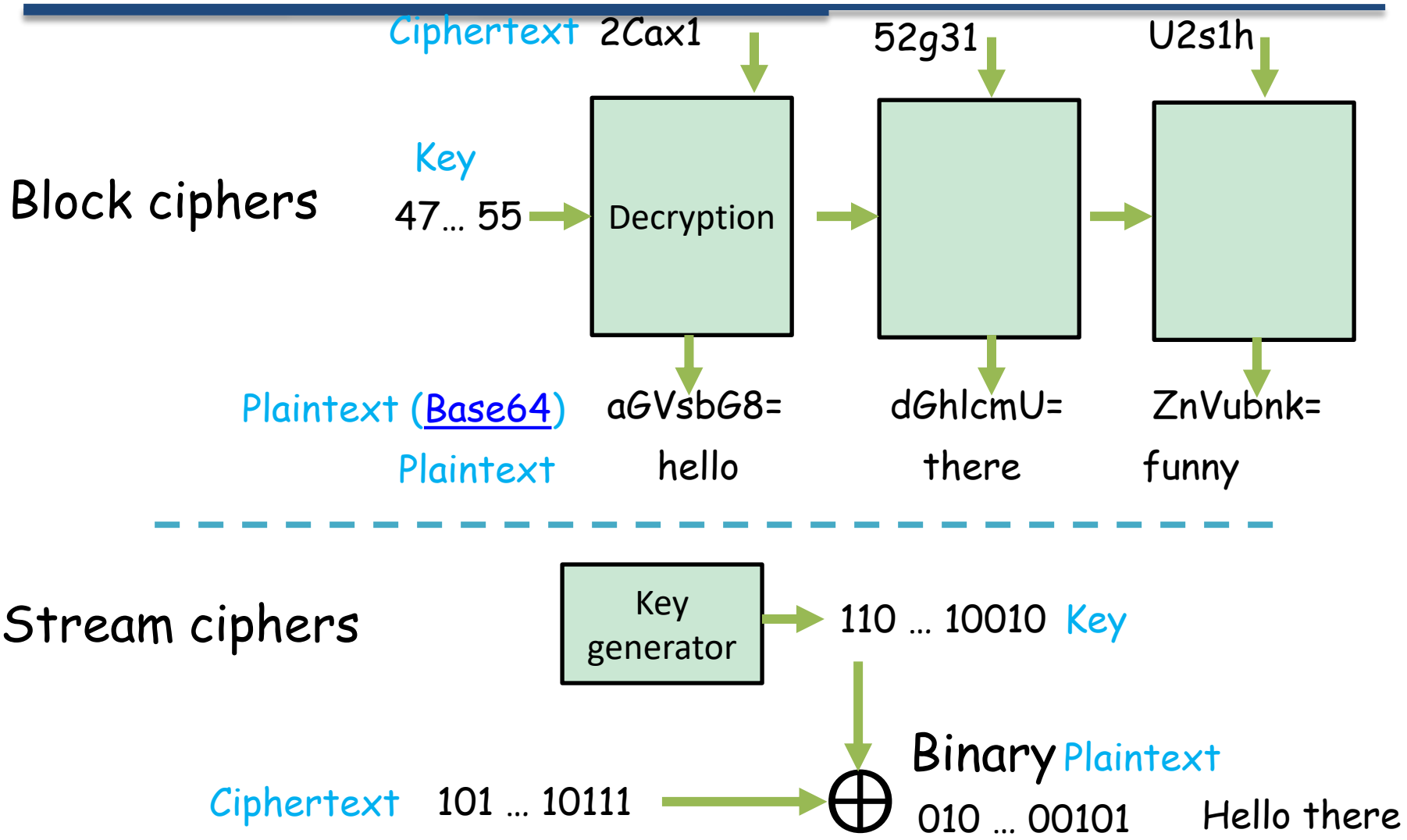
(a) Stream Cipher Using Algorithmic Bit Stream Generator

- Most ciphers are block ciphers
  - but it is possible to use a block cipher as a stream cipher (in some modes of operations that we will see later)

Example


Block ciphers
Plaintext
hello
there
funny
Key
47… 55
Encryption
Ciphertext
2Cax1
52g31
U2s1h
Stream ciphers
Key generator
110 … 10010 Key
Plaintext
Hello there
Binary Plaintext
010 … 00101
101 … 10111
Ciphertext

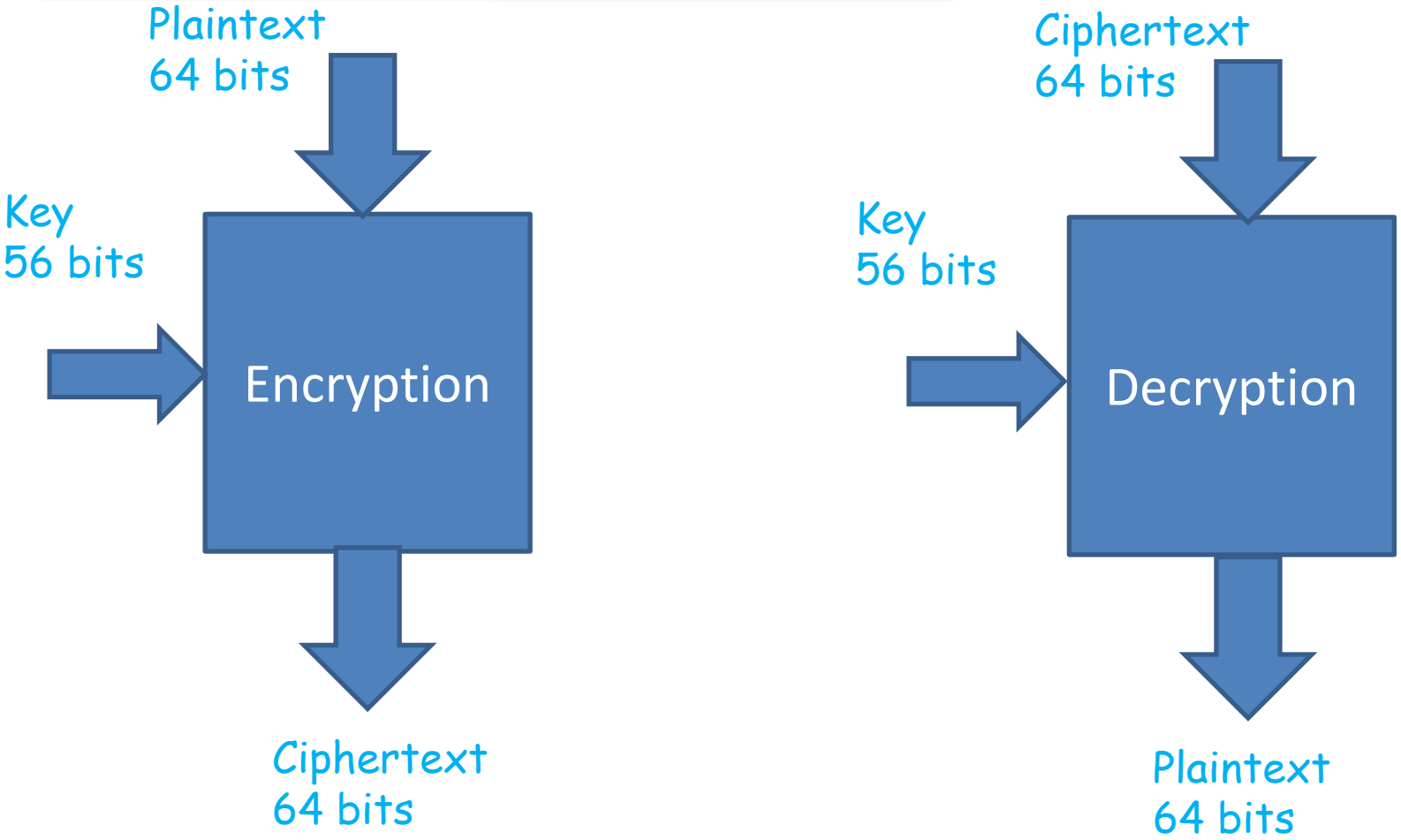
Example


Block ciphers
Ciphertext
2Cax1
52g31
U2s1h
Key
47… 55
Decryption
Plaintext (Base64)
aGVsbG8=
dGhlcmU=
ZnVubnk=
Plaintext
hello
there
funny
Stream ciphers
Key generator
110 … 10010 Key
Binary Plaintext
Ciphertext
101 … 10111
010 … 00101
Hello there

DES (Data Encryption Standard)

- DES was most widely used block cipher in world until recently.
- Adopted in 1977 by NBS (now NIST)
  - as FIPS PUB 46
- encrypts 64-bit data using 56-bit key
- Had widespread use
- There has been considerable controversy over its security

DES – Black box view



DES History

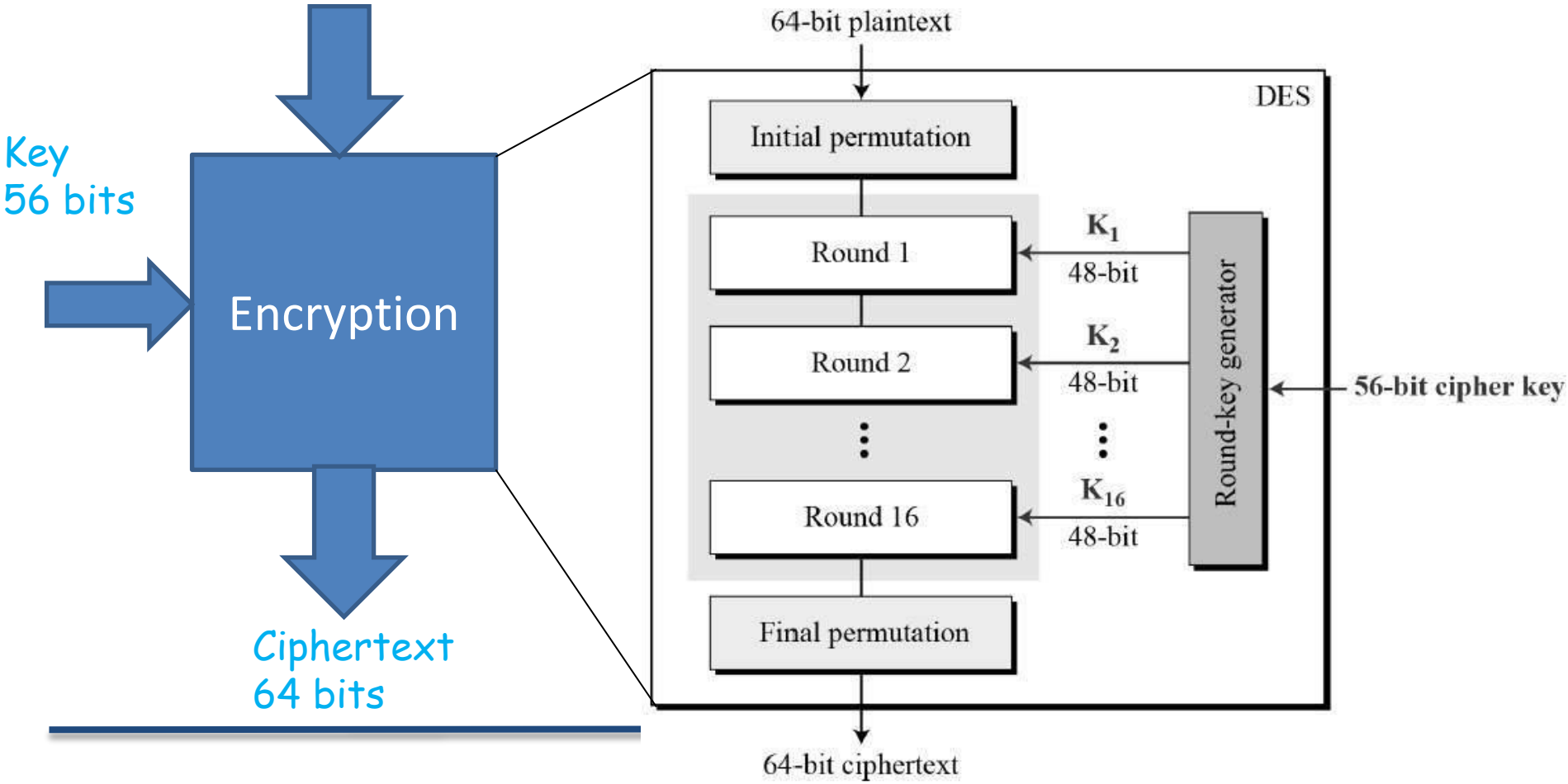
- IBM developed Lucifer cipher
  - by team led by Horst Feistel (1971)
  - used 64-bit data blocks with 128-bit key
- then redeveloped as a commercial cipher with input from NSA and others
- in 1973 NBS issued request for proposals for a national cipher standard
- IBM submitted their revised Lucifer which was eventually accepted as the DES
  - 56-bit key size!
- recertified in 1983, 1987 and 1993
- 3-DES (triple DES) has been issued as a new standard in 1999

DES Controversy

- Controversy over design
  - in choice of 56-bit key (vs Lucifer 128-bit)
  - design criteria (of the S-boxes) were classified
- S-boxes were fine
- but 56-bits became problem for DES as time goes by
  - due to advances in cryptanalysis and electronics
  - back in 1998 a project funded ($220K) by EFF (Electronic Frontier Foundation) broke DES in less than three days

Design of DES

- **DES** is a single combination of these techniques (a **substitution** followed by a **permutation**) on the plaintext

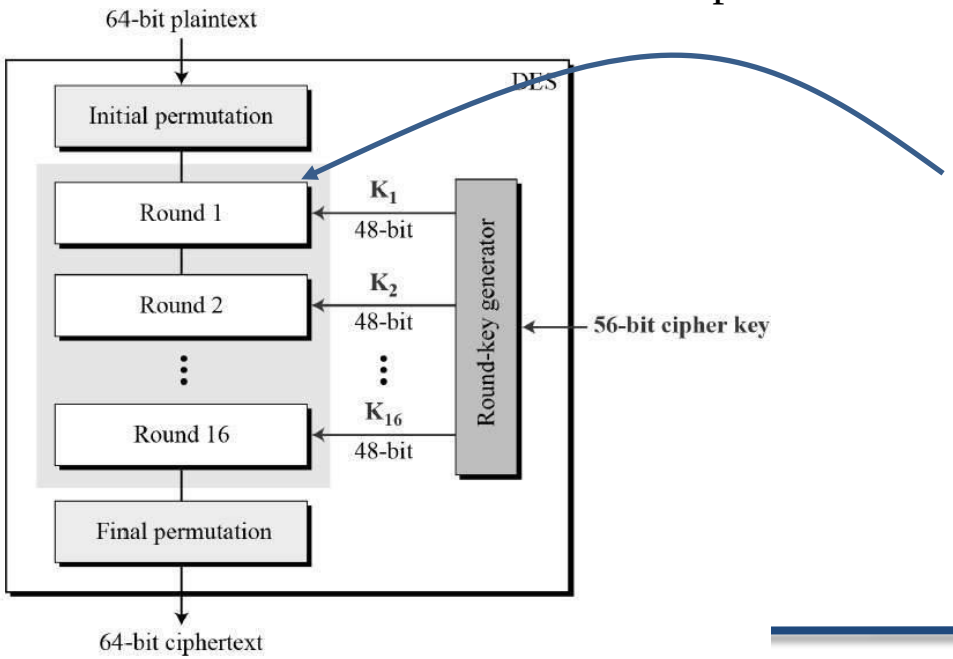


DES Characteristics

- DES is basically a product cipher
  - several rounds of substitutions and permutations
  - actually not that simple ☺ and you can see animation of DES here
- originally designed for hardware implementation
  - software implementations validated in 1993
  - but software DES is slow

DES Characteristics

- DES shows strong avalanche effect
  - one bit change in the input affects on average half of the output bits
  - to make attacks based on guessing difficult
- S-boxes are non-linear (substitution)
  - provides confusion
    - i.e. makes relationship between ciphertext and key as complex as possible



S-box

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	--	01	8D	F6	CB	52	7B	D1	E8	4F	29	C0	B0	E1	E5	C7
1	74	B4	AA	4B	99	2B	60	5F	58	3F	FD	CC	FF	40	EE	B2
2	3A	6E	5A	F1	55	4D	A8	C9	C1	0A	98	15	30	44	A2	C2
3	2C	45	92	6C	F3	39	66	42	F2	35	20	6F	77	BB	59	19
4	1D	FE	37	67	2D	31	F5	69	A7	64	AB	13	54	25	E9	09
5	ED	5C	05	CA	4C	24	87	BF	18	3E	22	F0	51	EC	61	17
6	16	5E	AF	D3	49	A6	36	43	F4	47	91	DF	33	93	21	3B
7	79	B7	97	85	10	B5	BA	3C	B6	70	D0	06	A1	FA	81	82
8	83	7E	7F	80	96	73	BE	56	9B	9E	95	D9	F7	02	B9	A4
9	DE	6A	32	6D	D8	8A	84	72	2A	14	9F	88	F9	DC	89	9A
A	FB	7C	2E	C3	8F	B8	65	48	26	C8	12	4A	CE	E7	D2	62
B	0C	E0	1F	EF	11	75	78	71	A5	8E	76	3D	BD	BC	86	57
C	0B	28	2F	A3	DA	D4	E4	0F	A9	27	53	04	1B	FC	AC	E6
D	7A	07	AE	63	C5	DB	E2	EA	94	8B	C4	D5	9D	F8	90	6B
E	B1	0D	D6	EB	C6	0E	CF	AD	08	4E	D7	E3	5D	50	1E	B3
F	5B	23	38	34	68	46	03	8C	DD	9C	7D	A0	CD	1A	41	1C

Other Important Symmetric Ciphers

- AES (Rjindael)
- 3DES (Triple DES)
- Blowfish
- RC5
- IDEA
- RC4

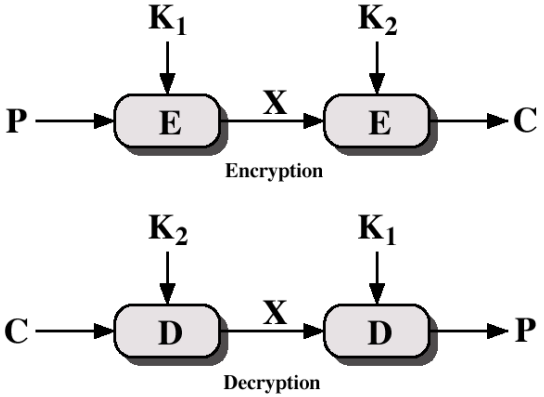
What happened after DES

- Replacement for DES was needed
  - vulnerability to cryptanalysis and practical brute-force attacks
- AES is the new standard (will see)
  - But took some time to standardize and deploy
- Meanwhile, some other ciphers are also used in practice (will briefly discuss too)
- But we still needed an immediate replacement of DES that can be standardized and deployed easily
  - This was 3DES

3DES (Triple DES)

- Another method for a strong cipher
- use multiple encryption with DES with different keys
  - to preserve the investment in DES
  - for quicker deployment
- Triple DES is chosen as a standard method
  - Standardized by ANSI, ISO and NIST

Why not double DES?

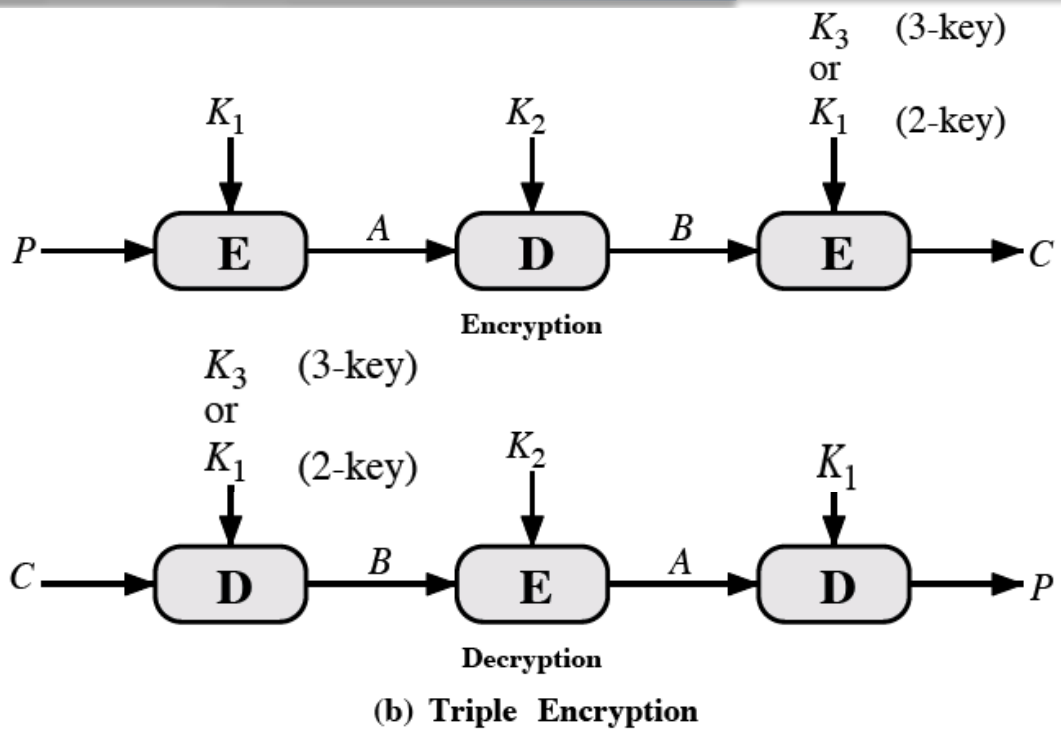


- Double DES
  - use DES two times with two different keys
  - Does not work due to meet-in-the-middle attack (which is a known-plaintext attack)
    - $X = E_{K1}[P] = D_{K2}[C]$
    - Try all possible K1's on P to create all possible X's and store them sorted
    - Try all possible K2's on C and match with above table
    - may create some false-alarms, so do the same attack for another plaintext-ciphertext pair
    - If the same K1-K2 pairs match for the second plaintext-ciphertext pair, then the correct keys are most probably found
    - complexity of this attack is close to the complexity of the single-DES brute-force attack, so double-DES is useless

3DES (Triple-DES)

- Three stages of DES
  - with two different keys
    - some attacks are possible but impractical
    - Merkle and Hellman, 1981
      - 2^{56} trials, but requires 2^{56} plaintext-ciphertext pairs
    - Oorschot and Wiener, 1990
      - $2^{120}/n$ trials, where n is the number of plaintext-ciphertext pairs
  - with three different keys
    - Attack complexity increases and becomes impractical

Triple-Des with two/three keys



(b) Triple Encryption

- E-D-E sequence
  - use of decryption at the second stage does not reduce/increase the security
  - Why decryption in the middle stage?

Triple-DES with three keys

- For those who feel some concern about the attacks on two-key 3-DES
- E-D-E sequence

$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$

- has been adopted by some Internet applications, eg PGP, S/MIME

Blowfish

- Developed by Bruce Schneier
  - author of the book *Applied Cryptography*
- 64-bit of block size
- Key size is variable
  - one to fourteen 32-bit blocks
    - 32 to 448 bits
    - provides a good trade-off between security and performance
- Fast and compact
- Has been implemented in numerous products
  - including GnuPG, SSH
  - see http://www.schneier.com/blowfish-products.html
- no known practical security problems

RC symmetric-key encryption algorithms

- Set of symmetric-key encryption algorithms invented by Ron Rivest who is also co-inventor of RSA cryptosystem.
- The "RC" may stand for either Rivest's cipher or, more informally, Ron's code.

– RC1 was never published.
– RC2 was a 64-bit block cipher developed in 1987.
– RC3 was broken before ever being used.
– RC4 is the world's most widely used stream cipher.
– RC5 is a 32/64/128-bit block cipher developed in 1994.
– RC6, a 128-bit block cipher based heavily on RC5, was an AES finalist developed in 1997.

RC5

- Ron's Code 5
  - developed by Ron Rivest who is also co-inventor of RSA cryptosystem
- owned and extensively used by RSA Inc.
- highly parametric
- word oriented processing that uses primitive operations that can be found in instruction sets of almost all microprocessors

RC5-w/r/b

- RC5 is actually a family of algorithms
- Parameters: w, r, b
  - w: Word size
    - 16, 32 or 64 bits
    - block size is 2*w
  - r: Number of rounds
    - 0 .. 255
  - b: key size in octets
    - 0 .. 255
- RC5 as suggested by Rivest is
  - RC5-32/12/16
  - 32-bit words (i.e. 64 bit blocks), 12 rounds, 128-bit key size

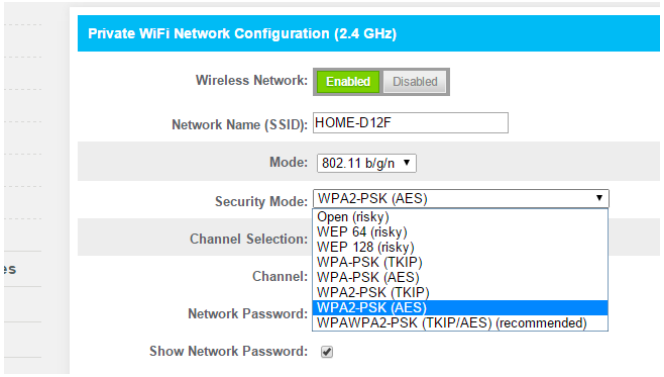
IDEA

- International Data Encryption Algorithm
- Lai and Massey of ETH Zurich (Swiss Federal Institute of Technology), 1990/91
- 64-bit blocks, 128-bit key size
- One of the early 128-bit algorithms
  - not US originated, so no export restrictions
  - used widely in PGP

- The original IDEA 8.5 round was broken in 2011, but a new version IDEA NXT is still valid.

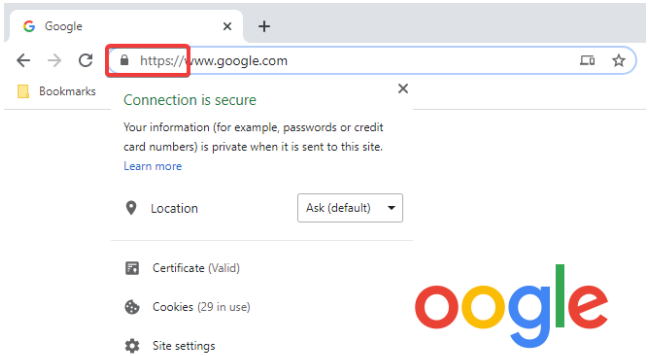
AES (Advanced Encryption Standard)

- Replacement needed for DES
  - reasons discussed before
- 3DES is a solution, but temporary
  - 3DES is slow in software
  - 3DES uses small blocks that makes even slower
- Need a new standard cipher
- AES was the replacement of DES and the most used encryption algorithm nowadays.



wifi settings

VPN services



Browse secure websites

AES Events in Chronological Order

- NIST issued call for a standard cipher in 1997
  - international
- 15 candidates (out of 21) accepted in June 98
- A shortlist of 5 selected in August 99
- Rijndael (from Belgium) was selected as the AES in October 2000
- issued as FIPS PUB 197 standard in November 2001

AES Requirements

- private key symmetric block cipher
- 128-bit data (block size)
- 128/192/256-bit keys
- stronger & faster than Triple-DES
- active life of 20-30 years
- provide full specification and design details

5 AES candidates

- MARS (IBM)
- RC6 (USA)
- Rijndael (Belgium)
- Serpent (Europe)
- Twofish (USA)

- Europe vs. USA
- commercial vs. academic
  - US based ones were all of commercial origin

AES Evaluation Criteria

- final criteria (used to select the winner)
  - general security
    - NIST relied on evaluation done by cryptographic community
  - software implementation performance
    - execution speed, performance across different platforms (8 to 64 bit platforms)
  - hardware implementation
    - not only timings, but also cost is important
    - especially for restricted space environments (such as smartcards)
  - implementation (timing and power) attacks

The AES Cipher - Rijndael

- designed by Vincent Rijmen and Joan Daemen in Belgium (UCL)
- has 128/192/256 bit keys, 128 bit block size
- Characteristics
  - resistant against known attacks
  - speed and code compactness on many platforms
  - design simplicity

Summary of Block Ciphers

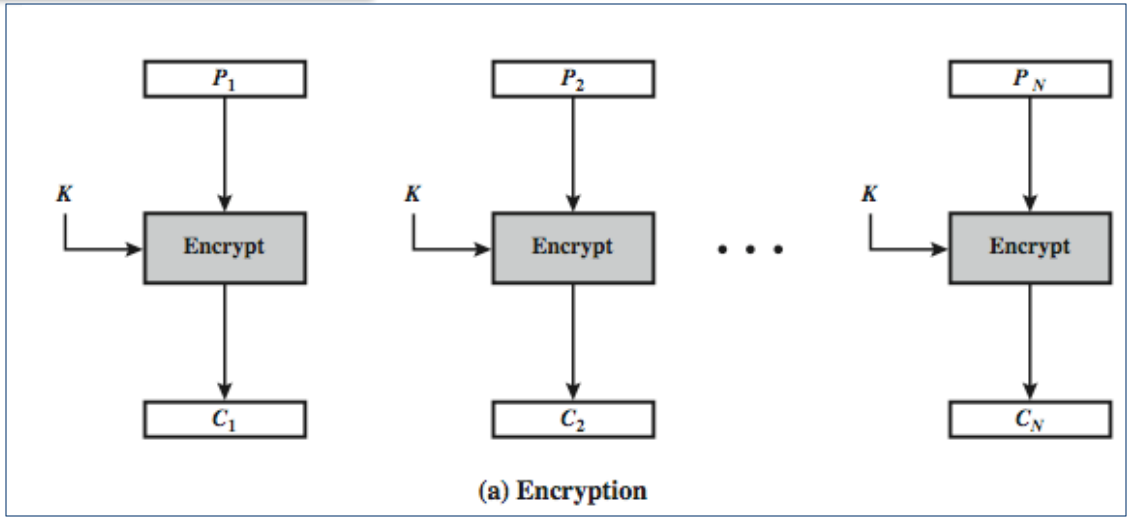
Algorithm (year)	Key size	Plaintext size	Ciphertext size	Security
DES (1971)	56 bits	64		Insecure
3DES (1995)	168, 112	64		Insecure
Blowfish (1993)	32–448	64		Safe enough
Twofish (1998)	128, 192 or 256	128		Secure
RC5 (1994)	(0 to 2040)128	32, 64,128(64)		Safe enough
RC6 (1998)	128, 192, or 256	128		Secure
IDEA (1991)	128	64		Insecure
AES (1998)	128, 192 or 256	128		Safe enough

Modes of Operations

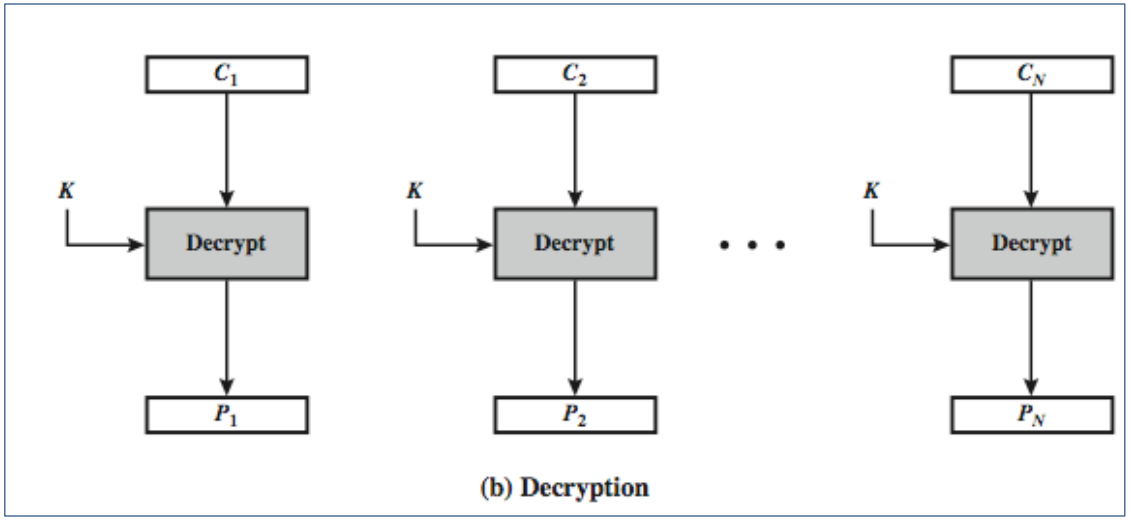
- block ciphers encrypt fixed size blocks
  - DES and 3DES encrypt 64-bit blocks
  - AES uses 128-bit blocks
- in practise, we have arbitrary amount of information to encrypt
  - we use DES, 3DES, AES and other symmetric ciphers in different modes in order to apply to several data blocks
- NIST SP 800-38A defines 5 modes
  - can be used with any block cipher

Electronic Codebook (ECB) Mode

- each block is encrypted independent of the other blocks
  - using the same key
- not so secure for long messages due to repetitions in code



(a) Encryption



(b) Decryption

ECB Mode

- Notation: C = E(P,K)
- Given plaintext $P_0,P_1,\ldots,P_m,\ldots$
- Most obvious way to use a block cipher:

Encrypt	**Decrypt**
$C_0 = E(P_0, K)$	$P_0 = D(C_0, K)$
$C_1 = E(P_1, K)$	$P_1 = D(C_1, K)$
$C_2 = E(P_2, K)$ …	$P_2 = D(C_2, K)$ …

- For fixed key K, this is "electronic" version of a codebook cipher (without additive)
  - With a different codebook for each key

ECB Cut and Paste

- Suppose plaintext is

  `Alice digs Bob. Trudy digs Tom.`

- Assuming 64-bit blocks and 8-bit ASCII:

  P_0 = "`Alice di`", P_1 = "`gs Bob. `",

  P_2 = "`Trudy di`", P_3 = "`gs Tom. `"

- Ciphertext: C_0, C_1, C_2, C_3
- Trudy cuts and pastes: C_0, C_3, C_2, C_1
- Decrypts as

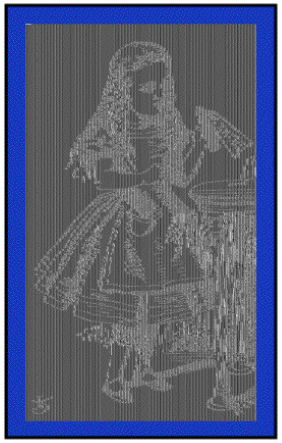
  `Alice digs Tom. Trudy digs Bob.`

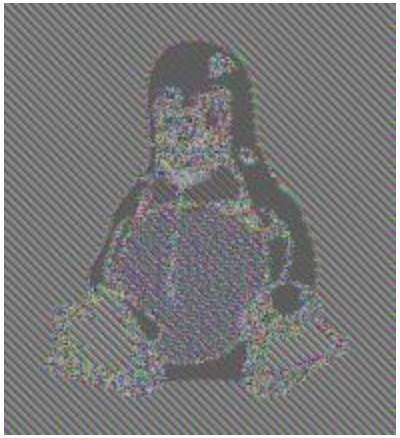
ECB Weakness

- Suppose $P_i = P_j$
- Then $C_i = C_j$ and Trudy knows $P_i = P_j$
- This gives Trudy some information, even if she does not know P_i or P_j
- Trudy might know P_i
- Is this a serious issue?

Alice Hates ECB Mode

- Alice's uncompressed image, and ECB encrypted

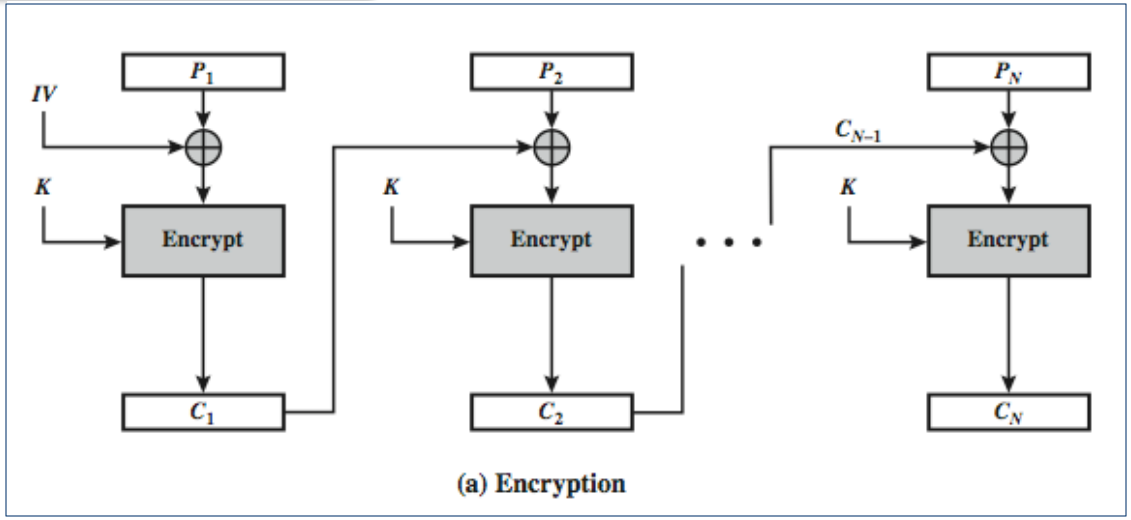



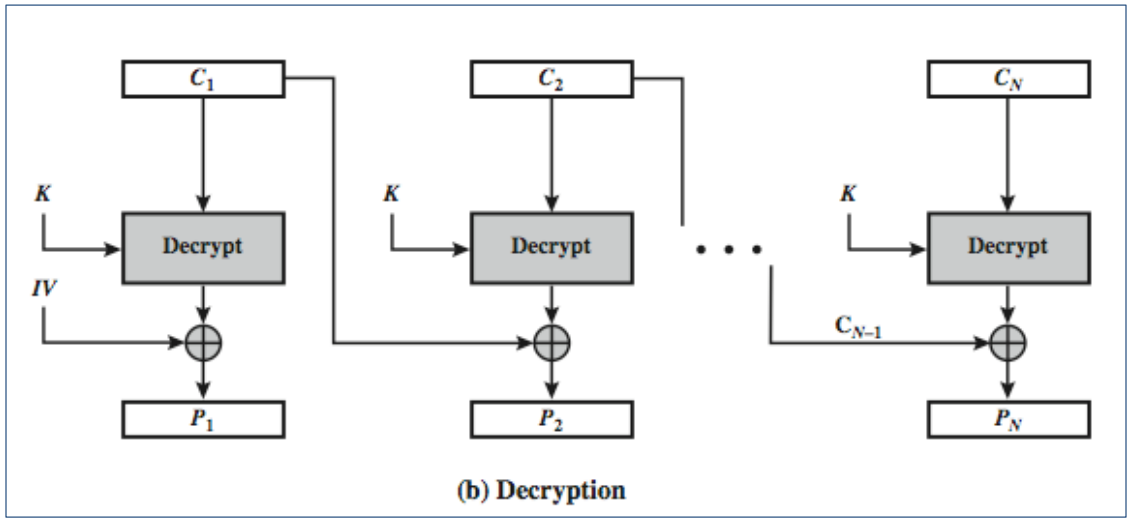


- Why does this happen?
- Same plaintext yields same ciphertext!

Cipher Block Chaining (CBC)

- each previous cipher blocks is XORed with current plaintext
- each ciphertext block depends on all previous blocks
- need Initialization Vector (IV) known to sender & receiver



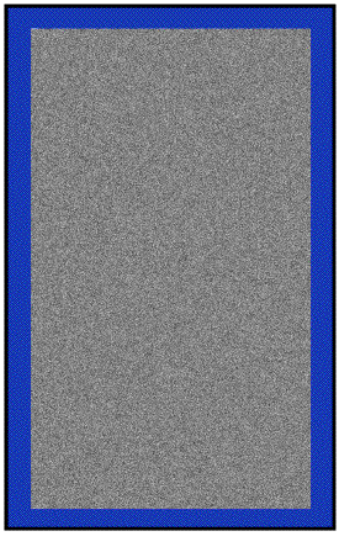


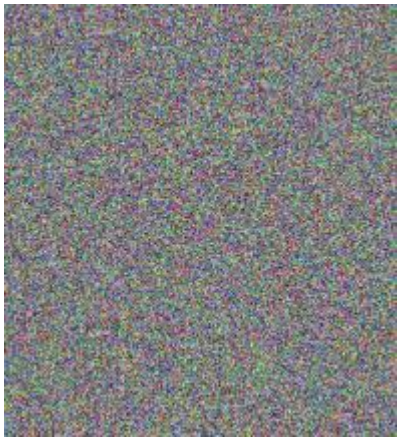
Cipher Block Chaining (CBC)

- Initialization Vector (IV)
  - both parties should agree on an IV
  - for maximum security, IV should be protected for unauthorized changes
  - Otherwise, attacker's change in IV also changes the decrypted plaintext
    - let's see this on board

Alice Likes CBC Mode

- Alice's uncompressed image, Alice CBC encrypted

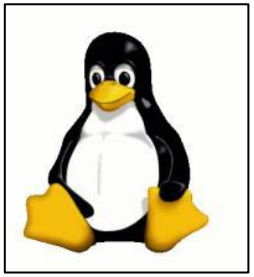


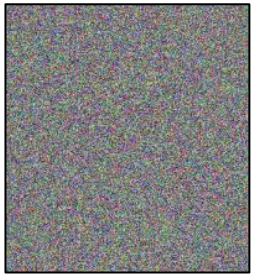
- Why does this happen?
- Same plaintext yields different ciphertext!

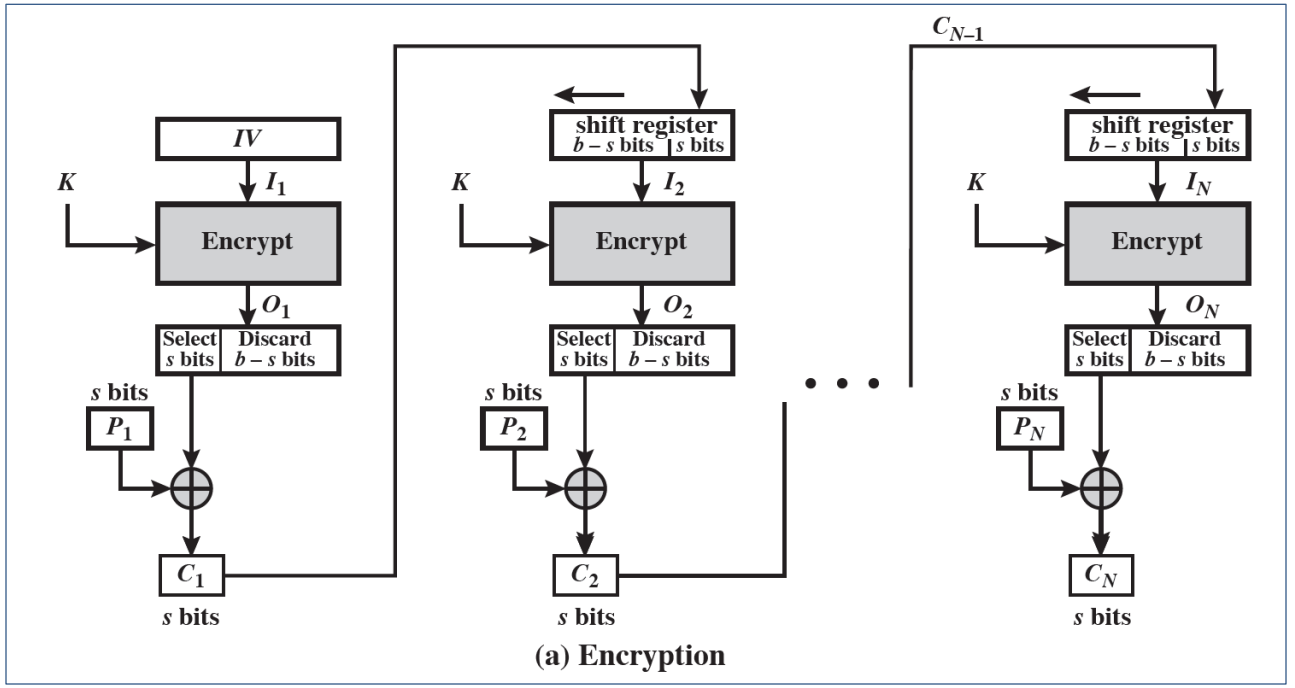
Cipher FeedBack (CFB)

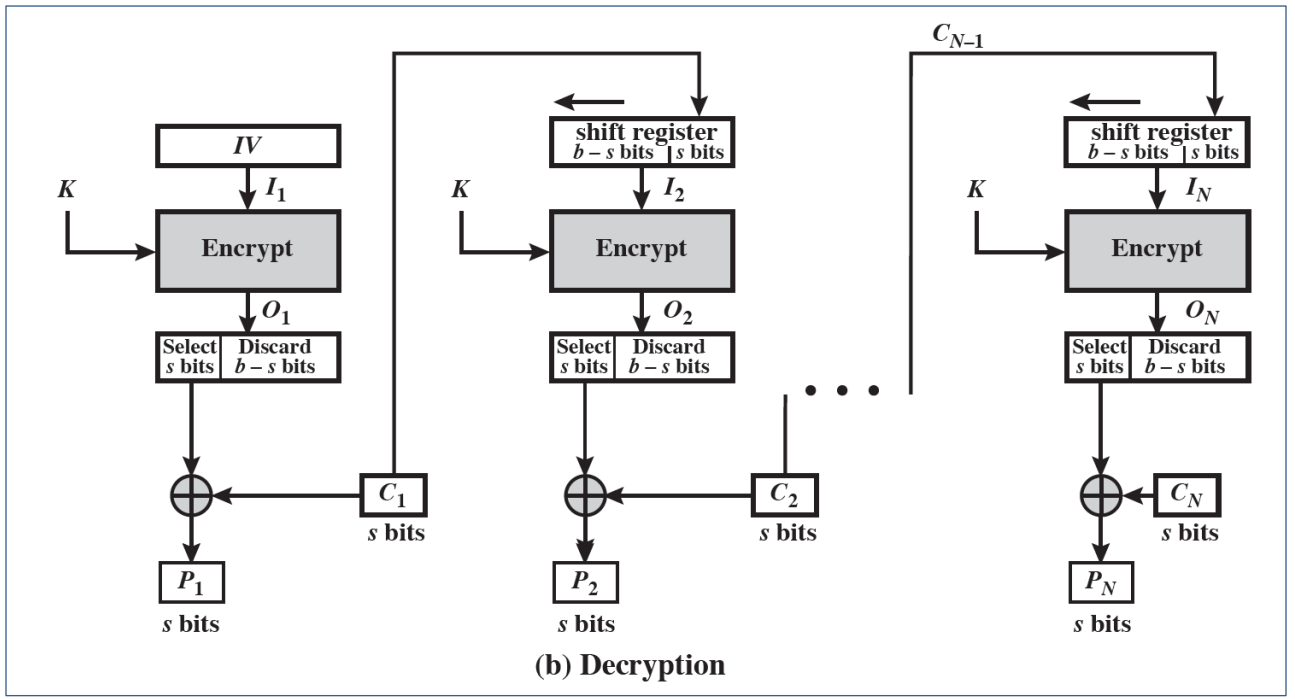
- Message is treated as a stream of bits
  - DES, AES (or any other block cipher) is used as a stream cipher
- standard allows any number of bit, *s*, (1,8 or more until the block size) as the unit of encryption/decryption
  - But common value for *s* is 8.
  - Plaintext is divided into block of *s* bits.
- uses IV
  - as all other stream ciphers
- Result of encryption is fed back to the next stage
- transmission errors propagate

Cipher FeedBack (CFB) Mode





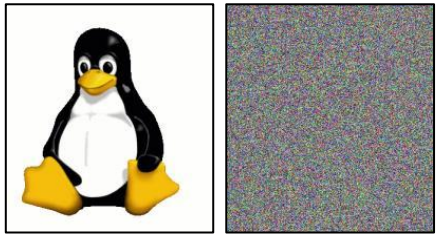


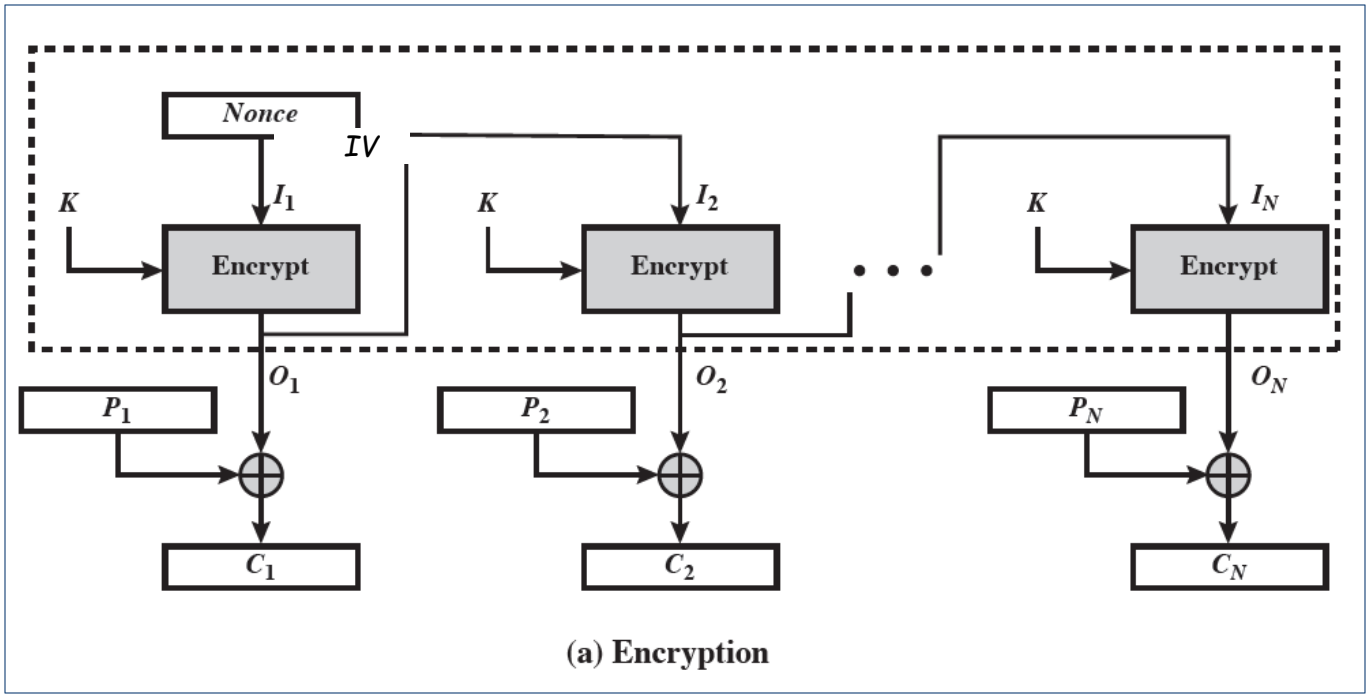


Output FeedBack (OFB)

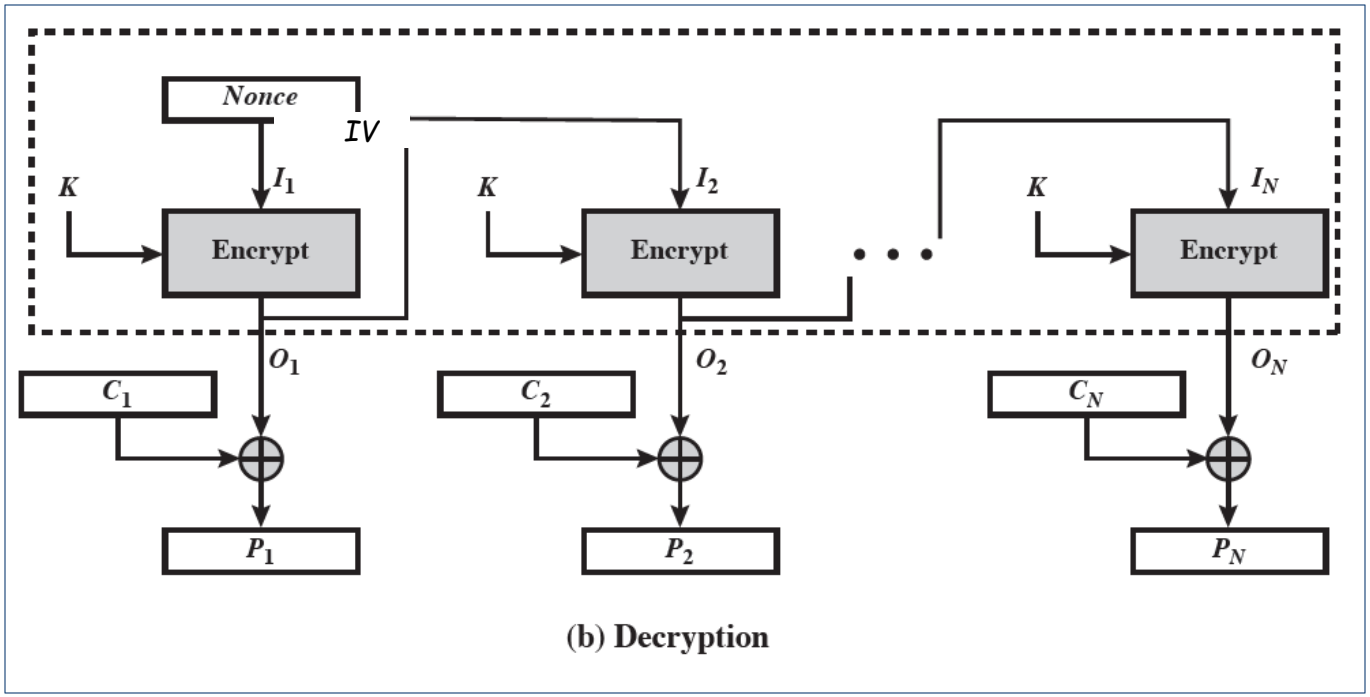
- another stream mode
  - but, *s*-bit version does not exist anymore
    - Full block is used in the encyption and decryption
- output of cipher is
  - XORed with the message
  - it is also the feedback
- feedback is independent of transmission, so transmission errors do not propagate
- same IV should not be used twice for the same key (general problem of using IV)
  - otherwise, when two ciphertext blocks are XORed the random sequence is cancelled and the attacker obtains XOR of two plaintexts
  - That is why IV is sometimes called as *nonce* (means "*used only once*")

Output FeedBack (OFB)





(a) Encryption

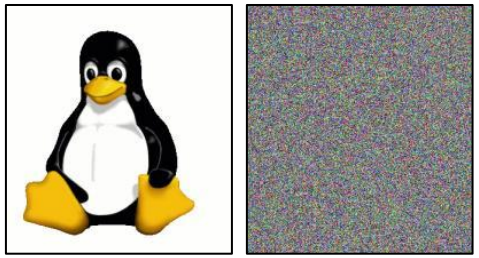


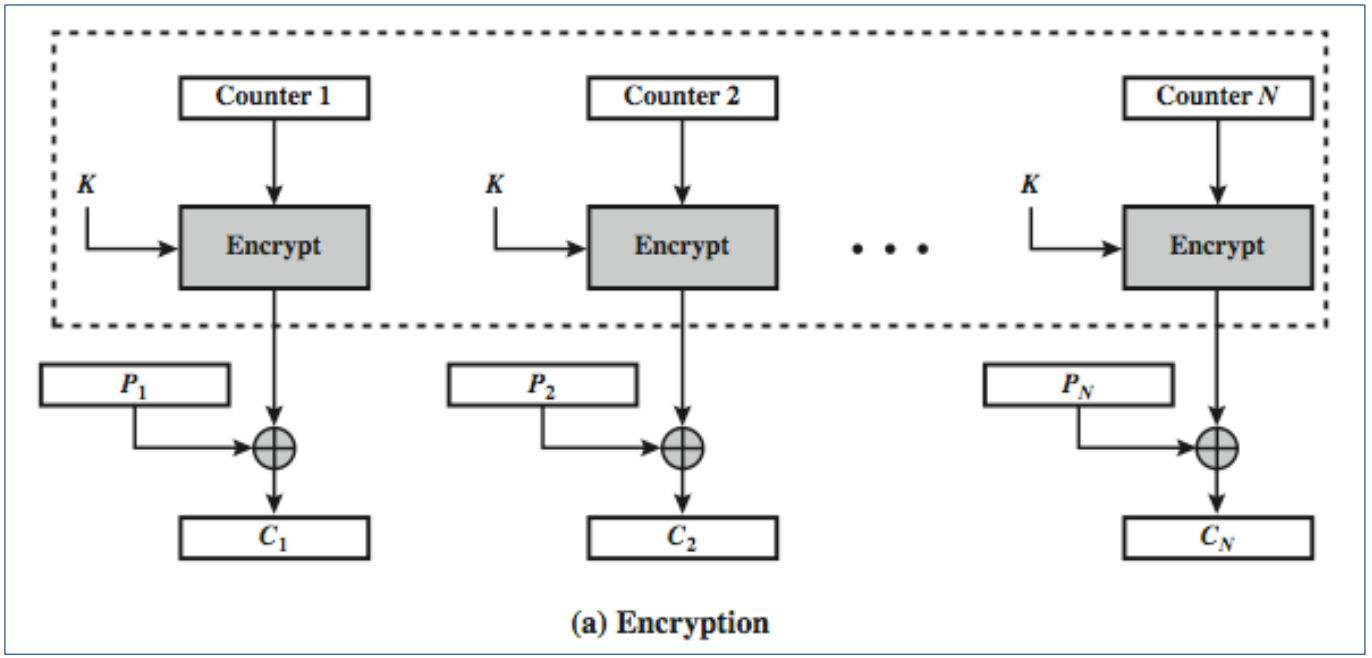
(b) Decryption

Counter (CTR)

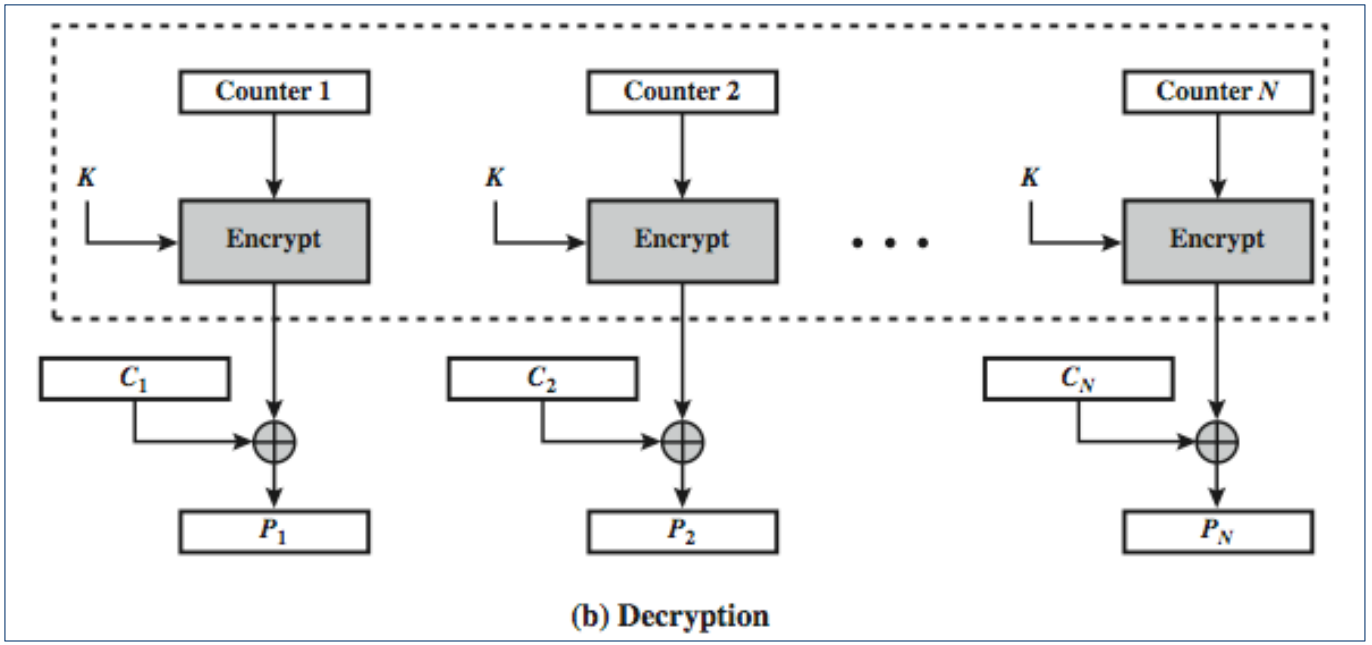
- similar to OFB but encrypts counter value rather than any feedback value
- For the same key, the counter value should not repeat
  - same problem as in OFB
- efficient
  - can do parallel encryptions
  - Cryptographic part of the process (encryption blocks) is performed in advance of need
  - good for bursty high speed links

Counter (CTR)





(a) Encryption



(b) Decryption

Random Numbers

- Many uses of *random numbers* in cryptography
  - nonces in authentication protocols to prevent replay
  - session keys
  - public key generation
  - keystream for stream ciphers

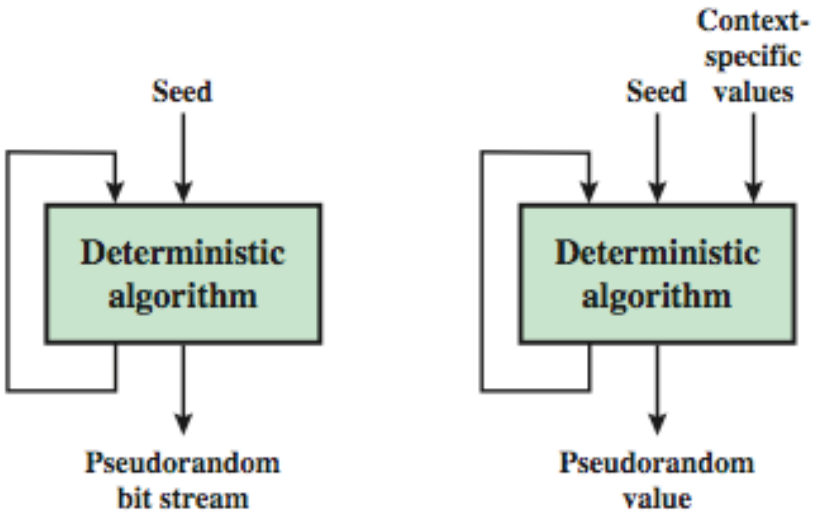
```
89206032161353150760
42991627100678658139
21768097580266432813
57585533115160214822
56368006290497803786
```

- Characteristics of random numbers
  - Statistical randomness
    - Uniform distribution of zeros and ones
    - İndependence of the bits in the sequence
  - Unpredictability of future values from previous values
- True random numbers provide these but very hard to obtain and use in practice

Pseudorandom Number Generators (PRNGs)

- often use deterministic algorithmic techniques to create "random numbers"
  - although are not truly random
  - can pass many tests of "randomness"
- known as "pseudorandom numbers"
- created by "Pseudorandom Number Generators (PRNGs)"

Pseudorandom Number Generators & Psuedorandom Functions



(b) PRNG (c) PRF

- Not much different
  - PRNG output is open-ended while PRF generates fixed size output
  - PRNG is mostly context independent while PRF is context dependent
  - Both may use feedback (there are some non-feedback ones too)
- When used in a cryptographic operation, seed must be kept secret

PRNG/PRF Requirements

- Randomness
  - Uniformity: the occurrence of zeros and ones must be equally likely
  - Scalability: any subseqeunce must pass randomness tests as well
  - Consistency: must not dependent on a particular seed value
- Unpredictability
  - forward unpredictability (next bits cannot be learned using previous bits)
  - backward unpredictability (seed cannot be learned using PRN sequence)
- There are some standard tests (total 15 of them) to check randomness and unpredictability (NIST SP800-22)
- Characteristics of the seed
  - secure
  - if known adversary can determine output
  - so must be random or pseudorandom number (there are some other standard tests for seed randomness as well)

Linear Congruential Generator

- Common iterative technique using:

  $X_{n+1} = (aX_n + c) \bmod m$

  X_0 is the seed
- Given suitable values of parameters can produce a long random-like sequence
- Suitable criteria to have are:
  - function generates a full-period (all values between 0 and m-1)
  - generated sequence should appear random
- Note that an attacker can reconstruct sequence given a small number of values
  - So, not a secure mechanism

Using Block Ciphers as PRNGs

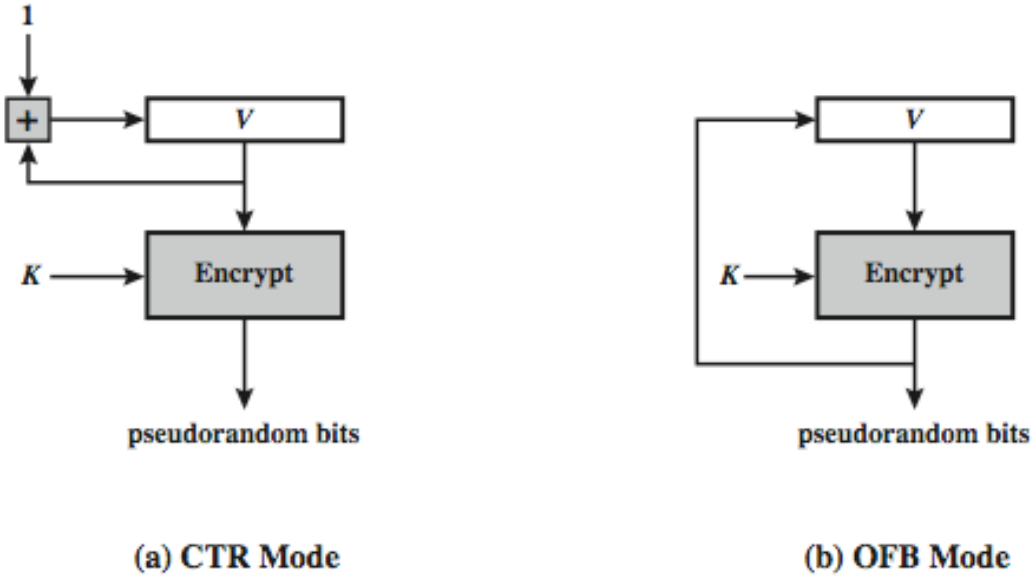
- for cryptographic applications, can use a block cipher to generate random numbers
- often for creating session keys from master key
- Standard methods
- CTR

$$X_i = E_K[V+i]$$

- OFB

$$X_i = E_K[X_{i-1}]$$

$$X_0 = E_K[V]$$

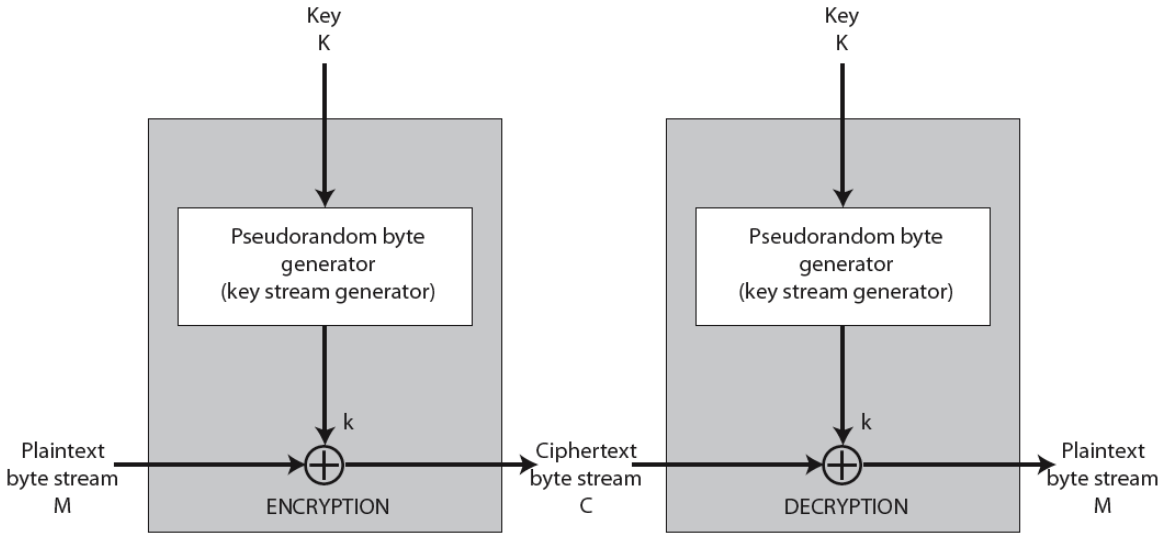


(a) CTR Mode (b) OFB Mode

(V, K) pair is the seed

Stream Ciphers

- process the message bit by bit
- Simply stating
  - a key and a Pseudo Random Number Generator (PRNG) is used to create a (pseudo) random key stream
  - keystream and the plaintext bitwise XORed to create the ciphertext
  - ciphertext is XORed with the same keystream to restore the plaintext



Some Stream Cipher Design Considerations

- A PRNG should eventually repeat
  - long period makes cryptanalysis difficult
- statistically randomness
  - e.g. approx. equal number of 0's and 1's
- large enough key (128-bit would be good to guard against brute-force attacks)

Stream Ciphers

- randomness of keystream destroys any statistical properties in the message
  - as in Vernam cipher and one-time pads
- Better than block ciphers in terms of
  - code space (implementations are simple)
  - throughput (faster per bit en/decryption)
- but must never use the same keystream more than once
  - otherwise the cryptanalyst can XOR two ciphertext streams and find out XOR of two plaintext streams
    - not so difficult to crack

Stream Ciphers

- are useful if data are transferred as a stream
  - web browser
  - voice
  - video
- actually any block cipher can be used as a stream cipher
  - CFB mode of operation (and OFB and CTR)

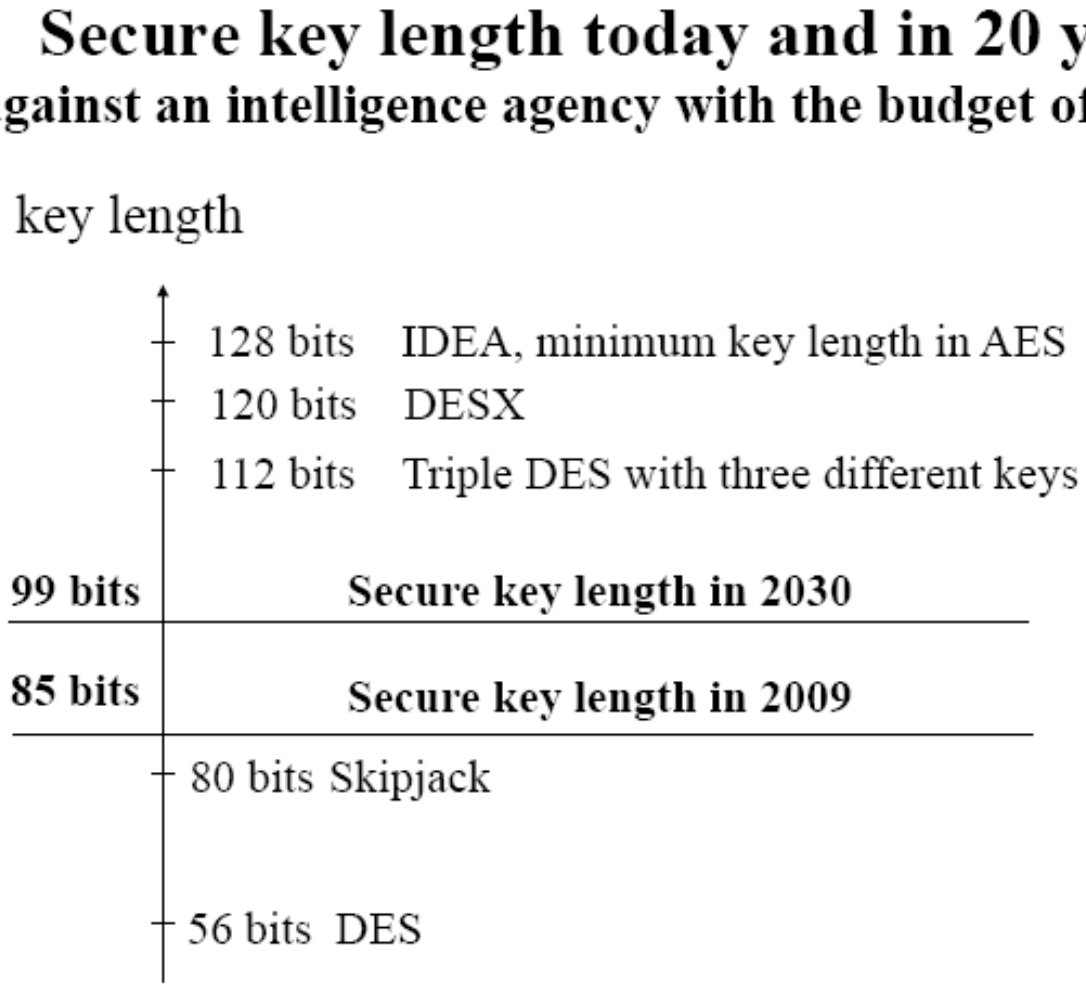
RC4

- Ron's Code 4
- Yet another cipher designed by Ron Rivest
  - owned by RSA Inc.
  - was kept as a trade secret, but in 1994 anonymously posted on the Internet
- variable key size, byte-oriented stream cipher
- simple but effective
  - 8 to 16 machine operations per output byte
- widely used (SSL/TLS, WEP/WPA)
- Some attacks reported, but not practical for key size greater than 128-bit
- However, WEP has a problem due to RC4 key generation
  - not a problem of RC4 in particular

and other symmetric ciphers

- CAST
- Skipjack
- Serpent
- Twofish
- RC6
- Mars
- SAFER+

Discussion



Courtesy of Kris Gaj

Discussion

- Assuming ~92-bit is secure enough for today and Moore's Law continues
  - 1 bit per 18 months to be added
    - 2020's: 93-bit (approx.)
    - 2040's: 107-bit (approx.)
  - with 128-bit, AES we will be secure for a long time
- unless a new efficient cryptanalysis method is found
  - known cryptanalysis methods are not practical for secure key sizes for 3DES, AES, IDEA, etc. (except DES of course)