Bachelor of Information System

IS2109 - Information System Security

Additional Lecture - 2

Kasun De Zoysa
kasun@ucsc.cmb.ac.lk

UNIVERSITY OF COLOMBO SCHOOL OF COMPUTING

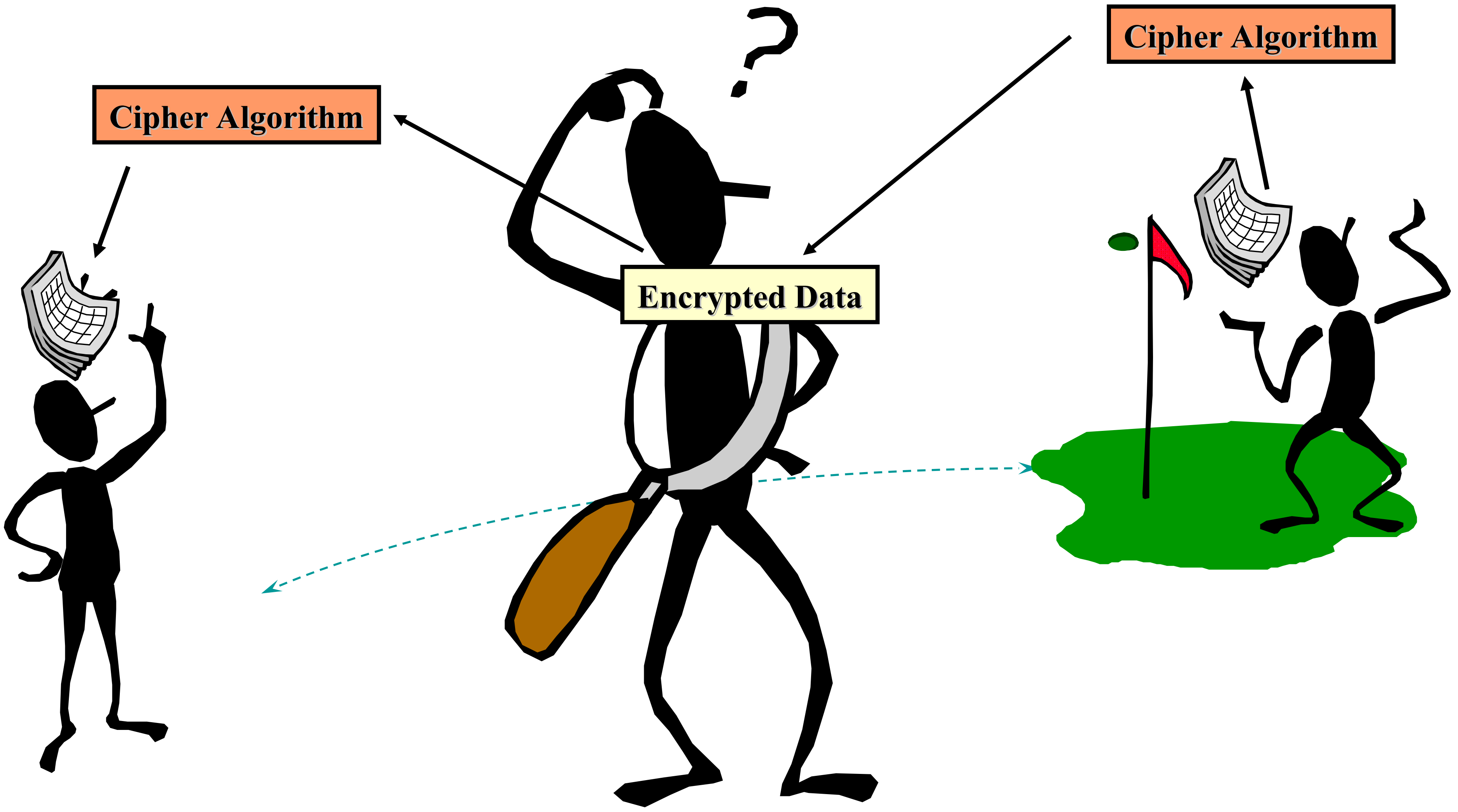
Objectives - Cryptography

The Cryptography domain addresses the <u>principles</u>, <u>means</u>, and <u>methods of disguising information</u> to ensure its integrity, confidentiality, authenticity and non-repudiation(?).

What You Should Know

- Basic concepts and terms within cryptography
  - Public and private key algorithms in terms of their applications and uses
  - Cryptography algorithm construction, key distribution, key management, and methods of attack
  - Applications, construction, and use of digital signatures
  - Principles of authenticity of electronic transactions and non-repudiation

Basic Concept



Definitions

- **Cryptography**
  - Art or science of secret writing
  - Protects sensitive information from disclosure
  - Storing and transmitting information in a form that allows it to be revealed only to those intended
  - Cryptosystem accomplishes this
  - Identifies the corruption or unauthorized change of information
  - Designed to make compromise too expensive or too time-consuming
- **Cryptanalysis**
  - art/science relating to converting ciphertext to plaintext without the (secret) key
  - descrambling without secret key ; art of breaking ciphers
  - Practice of defeating such attempts to hide info
- **Cryptology**
  - Includes both cryptography and cryptanalysis

Definitions , continued

- **Encipher**
  - act of scrambling

- **Decipher**
  - descrambling with secret key
- **Key**
  - secret sequence governing en/deciphering

Cryptography Basic

- **Why Encrypt?**
  - Protect stored information
  - Protect information in transmission
- Cryptography originally used for secrecy
- **Encryption** - process by which **plaintext** is converted to **ciphertext** using a **key**
- **Decryption** - process by which ciphertext is converted to plaintext (with the appropriate key)
- **plaintext** (cleartext)- intelligible data

Cryptography Business Use

:

- Prevent unauthorized disclosure of information
- Prevent unauthorized access to information, computers, web sites, applications,etc.
- Detect tampering
- Detect injection of false data
- Detect deletion of data
- Prevent repudiation

The goal of a cryptosystem

- **The goal of a cryptosystem is to provide**
- **Confidentiality** To ensure that unauthorized parties cannot access the data, message or information
- **Authenticity** To ensure that the source / sender of the data, message or information is identifiable
- **Integrity** To ensure that the data. Message or Information was not modified during transmission
- **Nonrepudiation** To ensure that either party cannot deny sending or receiving the data, message or information

Cryptography History

- **Historic examples...**
  - Earliest cryptography: an Egyptian scribe using non-standard hieroglyphics
  - Julius Caesar ("Caesar Cipher")
    Each plaintext letter is replaced by a letter some fixed number of positions further down the alphabet (e.g. Belgica (3 positions) → ehojlfd)
  - The Kama Sutra recommends cryptography as 44th and 45th art
    (of 64) men and women should know

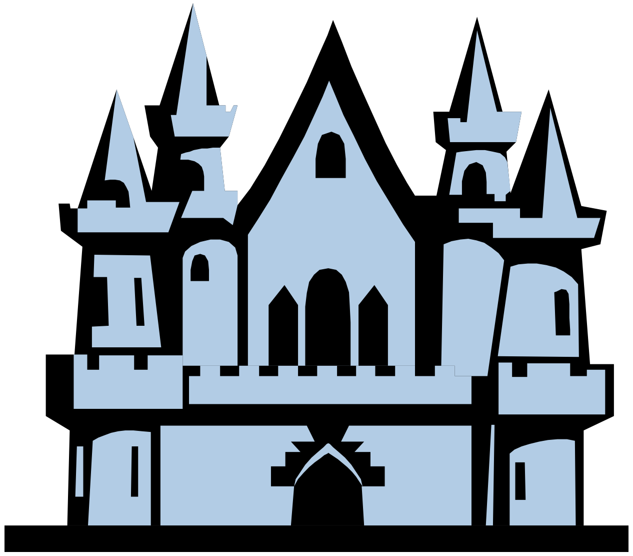
Cryptography History

- ENIGMA Used by the Germans in WW2 – and the subsequent code-breaking activities at Bletchley park (still a popular subject of books and movies)
- 1976: Public Key Cryptography concept (Whitfield Diffie & Martin Hellman)
- 1977: first (*published*) practical PKC cryptosystem invented (RSA - Rivest, Shamir, Adleman)
- October 2000 Rijndael is chosen as AES (Advanced Encryption Standard)

The Caesar Cipher

Plain Text	: 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
Cipher Text	: 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

$$C_i = E(P_i) = P_i + 3$$



Kamasutra

One of the earliest descriptions of encryption by substitution appears in the Kama-sutra, a text written in the 4th century AD by the Brahmin scholar Vatsyayana, but based on manuscripts dating back to the 4th century BC.

How it work

The kamasutra generate list of 26 alphabet with no duplicate. Then divide by 2 row. Find for each letter of message text in table and choose the opposite of the letter

Kamasutra

for example:

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

divide by 2 rows

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

Given String = KAMASUTRA

K is at 2nd row and 5th column. Get the opposite of K that is I. Do each letter until the end

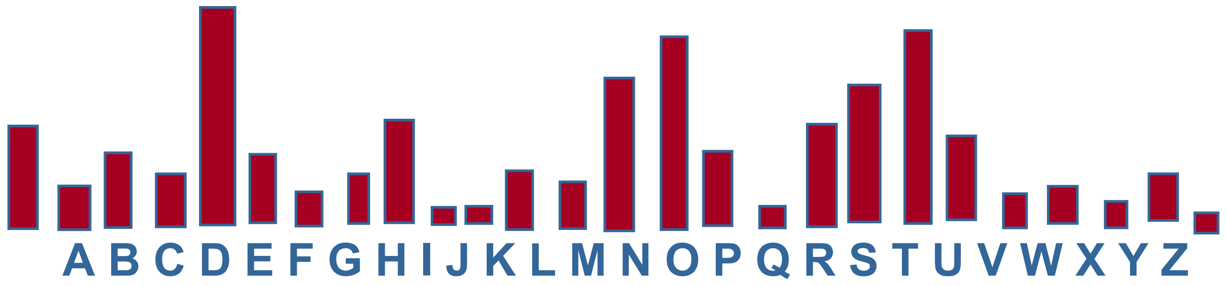
Cipher : IZOZNQJYZ

Monoalphabetic Substitutions

Plain Text : 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

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

Letter Frequency



One Time Pad / Vernam Cipher

- Invented in 1917 by Gilbert Vernam and Joseph Mauborgne.

  Usually implemented as a stream cipher using the XOR function.
- Key is used once and discarded by both sender and receiver.

  Length of the Key character stream is equal to the message length.
- Not practical for large amounts of data (MB / GB).
- Pad is theoretically unbreakable by exhaustive brute force.
- Implementation uses a Key that consists of a set of random
- non-repeating characters.
- Each Key letter and Plaintext are added modulo 26 to each other and then converted back into a letter.

One Time Pad / Vernam Cipher

Plain Text	: V E R N A M C I P H E R
Numeric Equivalent	: 21 4 17 13 0 12 2 8 15 7 4 17
+Random Number	: 76 48 16 82 44 3 58 11 60 5 48 88
= Sum	: 97 52 33 95 44 15 60 19 75 12 52 105
=Mod 26	: 19 0 7 17 18 15 8 19 23 12 0 1
Cipher text	: t a h r s p I t x m a b

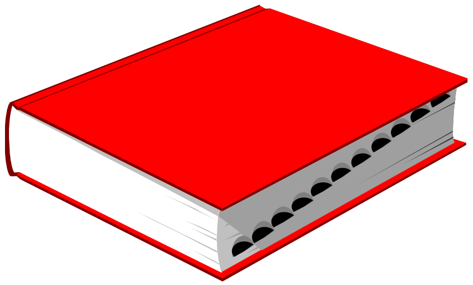
Binary Vernam Cipher

Plain Text	: 1 0 1 0 0 0 1 1 1 0 0 1 1 0 1
$\oplus$Random Stream	: 0 1 0 1 1 0 1 0 1 1 1 0 1 0 1
Cipher text	: 1 1 1 1 1 0 0 1 0 1 1 1 0 0 0

Random Numbers

1. Truly Random numbers

- **Books**
- **CD**



2. Pseudo Random numbers

- **Linear congruential random number generation**

$$\boxed{R_{i+1} = (a * R_i + b) \bmod n}$$

Encipherment Modes

- Stream Ciphers - Message broken into characters or bits and enciphered with a “key stream”
  - key stream - should be random and generated independently of the message stream

- Block ciphers process messages in blocks, each of which is then en/decrypted

Block vs Stream Ciphers

- Block ciphers process messages in blocks, each of which is then en/decrypted
- Like a substitution on blocks of characters
    - 64-bits or more

- Stream ciphers process messages a bit or byte at a time when en/decrypting
- E.g. Vernam cipher, one time pad

- Many current ciphers are block ciphers

Kerckhoff’s Principle

The security of the encryption scheme must depend only on *the secrecy of the key and not on the secrecy of the algorithms.*

Reasons:

•Algorithms are difficult to change

•Cannot design an algorithm for every pair of users

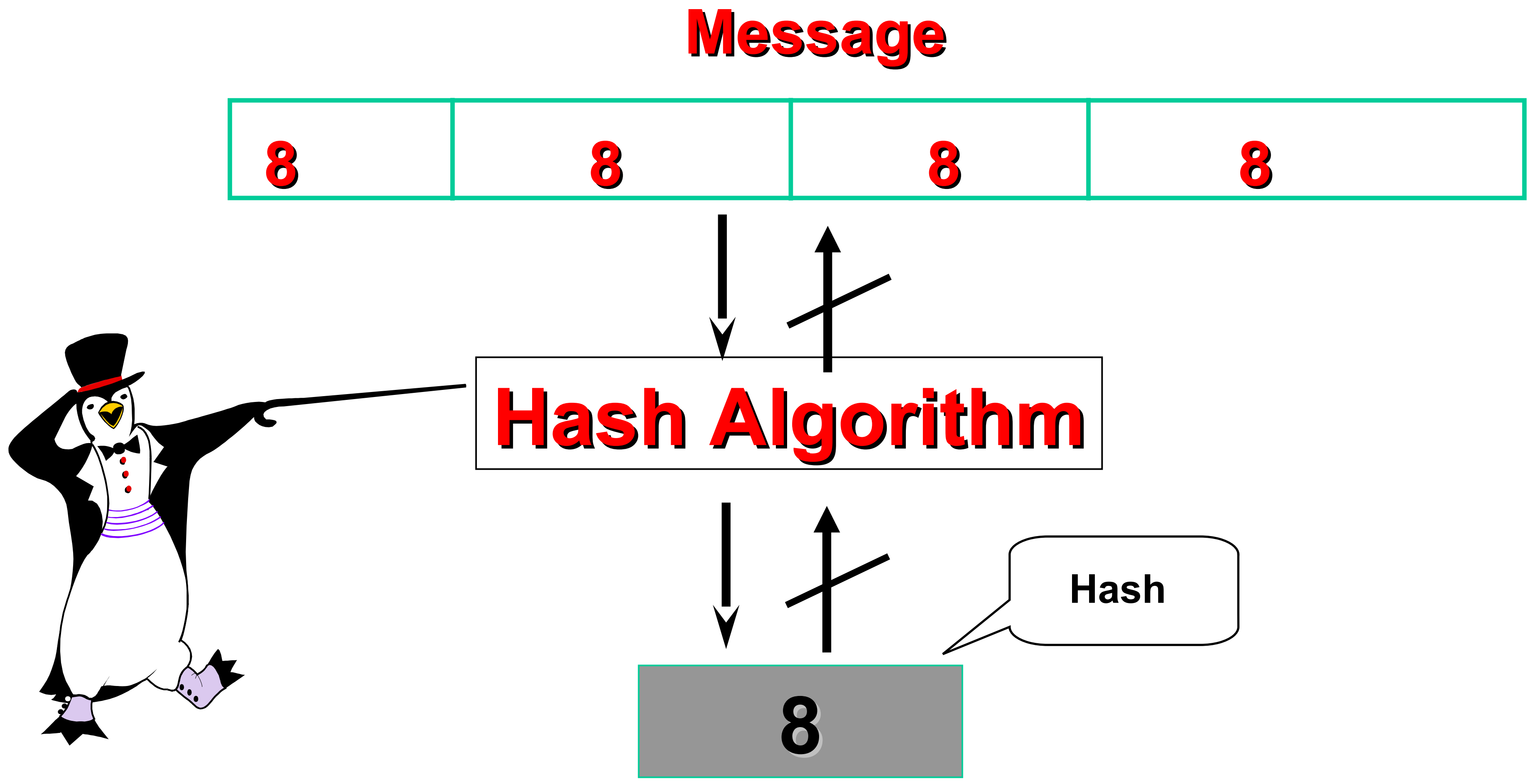
•Expert review

•No security through obscurity!

Hash Functions

- Condenses arbitrary message to fixed size
- Usually assume that the hash function is public and not keyed
  - MAC which is keyed (will discuss soon)
- Hash used to detect changes to message
- Can use in various ways with message
  - most often to create a password, digital signature etc.

Hash Functions



Simple Hash Functions

- There are several proposals for simple functions
- Some are based on XOR of message blocks
- Not secure since one can manipulate any message and either not change hash or manipulate the hash as well
- need a stronger cryptographic function

Hash Function Properties

•A Hash Function produces a fingerprint of some file/message/data

$$h = H(M)$$

•condenses a variable-length message M to a fixed-sized fingerprint

•Assumed to be public

Requirements for Hash Functions

- Can be applied to any sized message M
- Produces fixed-length output h
- Easy to compute h = H(M) for any message M
- Given h, it is infeasible to find x s.t. H(x) = h

  one-way property
- Given x, it is infeasible to find y s.t. H(y) = H(x)

  weak collision resistance
- It is infeasible to find any x,y s.t. H(y) = H(x)

  strong collision resistance

MD5

- Designed by Ronald Rivest (the R in RSA)
- Latest in a series of MD2, MD4
- Produces a 128-bit hash value
- Until recently was the most widely used hash algorithm
  - in recent times have both brute-force & cryptanalytic concerns
- Specified as Internet standard RFC1321

Strength of MD5

- MD5 hash is dependent on all message bits
- Rivest claims security is good as can be
- Known attacks are:
  - Berson (92) attacked any 1 round using differential cryptanalysis (but can't extend)
  - Boer & Bosselaers (93) found a pseudo collision (again unable to extend)
  - Dobbertin (96) created collisions on MD compression function (but initial constants prevent exploit)
  - Crypto 2004 attacks on SHA-0 and MD5
- Conclusion is that MD5 has been shown to be vulnerable
- MD5 Collision Demo: http://www.mscs.dal.ca/~selinger/md5collision/

Secure HASH Functions

- Purpose of the HASH function is to produce a ”fingerprint.
- Properties of a HASH function H :

1. H can be applied to a block of data at any size
2. H produces a fixed length output
3. H(x) is easy to compute for any given x.
4. For any given block x, it is computationally infeasible to find x such that H(x) = h
5. For any given block x, it is computationally infeasible to find with H(y) = H(x).
6. It is computationally infeasible to find any pair (x, y) such that H(x) = H(y)

Comparison - Modern HASH functions

Algorithm Name	**Comparison**
SHA-256	Widely used cryptographic hash function that generates a fixed-size output of 256 bits. It is considered secure and is commonly used in digital signatures, password storage, and other applications where data integrity and authenticity are critical.
Argon2	Relatively new hashing algorithm that is designed to be resistant to both brute-force and side-channel attacks. It is commonly used for password storage and key derivation, and is considered to be one of the most secure hashing algorithms available.
Yescrypt	Memory-hard hashing algorithm that is designed to be resistant to brute-force attacks and specialized hardware attacks, such as ASICs. It is commonly used for password storage and key derivation. Yescrypt is much slower than SHA-256, as it requires a large amount of memory to generate a hash value.

Hashing Applications

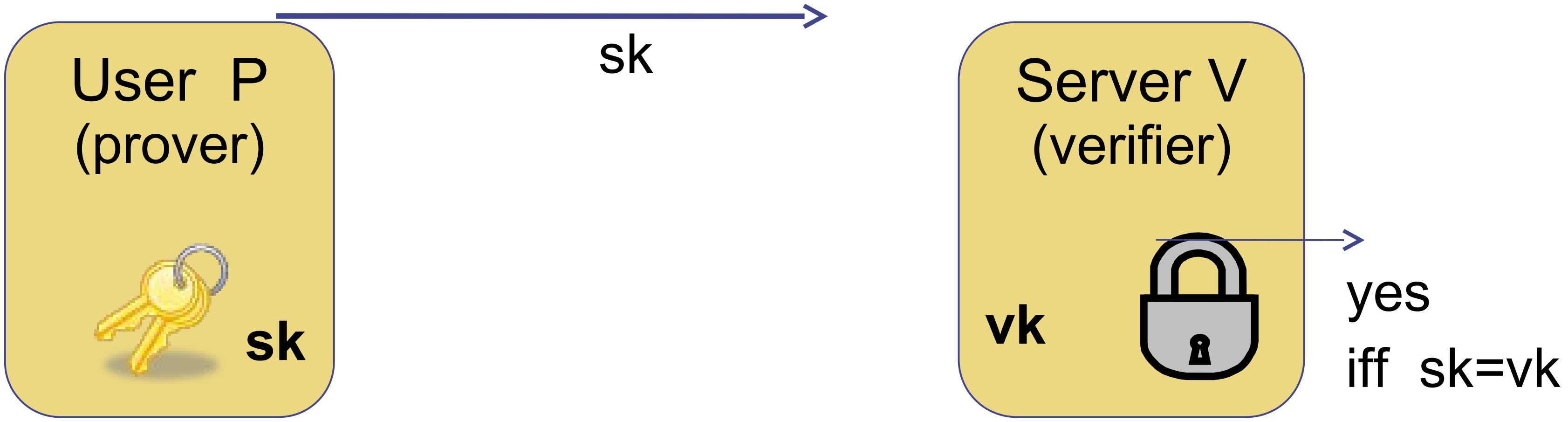
- **Password storage:** Hashing algorithms are commonly used to store passwords securely.
- **Message authentication:** Hashing algorithms are often used to authenticate messages and ensure their integrity.
- **Digital signatures:** Hashing algorithms are used in digital signature schemes to verify the authenticity of a digital document or message.
- **Key derivation:** Hashing algorithms are often used to derive cryptographic keys from a password or other secret value.

Hashing Applications

- **Data deduplication:** Hashing algorithms can be used to identify and remove duplicate data in storage systems.
- **Chain of custody (CoC)**:Hashing algorithms can be used for tracking process that is used to establish the chronological history of the custody of digital evidence.
- **Blockchain technology:** Hashing algorithms are a critical component of blockchain technology, which is used for secure and decentralized record-keeping.
- **Many more ..**

Basic Password Protocol (incorrect version)

- **PWD**: finite set of passwords
- Algorithm G (KeyGen):
  - choose rand pw in PWD. output sk = vk = pw.



Basic Password Protocol (incorrect version)

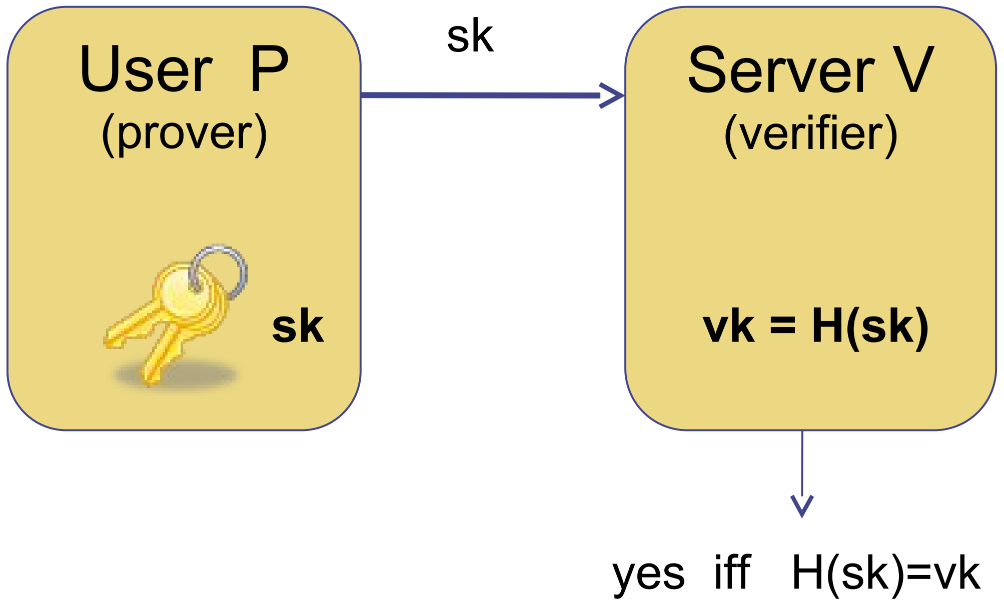
- Problem: VK must be kept secret
  - Compromise of server exposes all passwords
  - Never store passwords in the clear!

password file on server

Alice	pw_{alice}
Bob	pw_{bob}
…	…

Basic Password Protocol: version 1

H: one-way hash function from PWD to X
"Given H(x) it is difficult to find y such that H(y)=H(x)"



password file on server

Alice	$H(pw_A)$
Bob	$H(pw_B)$
…	…

Weak Passwords and Dictionary Attacks

- People often choose passwords from a small set:
  - The 6 most common passwords (sample of 32×10^6 pwds):
    123456, 12345, Password, iloveyou, princess, abc123
    ('123456' appeared 0.90% of the time)
  - 23% of users choose passwords in a dictionary of size 360,000,000

- **Online dictionary** attacks:
  - Defeated by doubling response time after every failure
  - Harder to block when attacker commands a bot-net

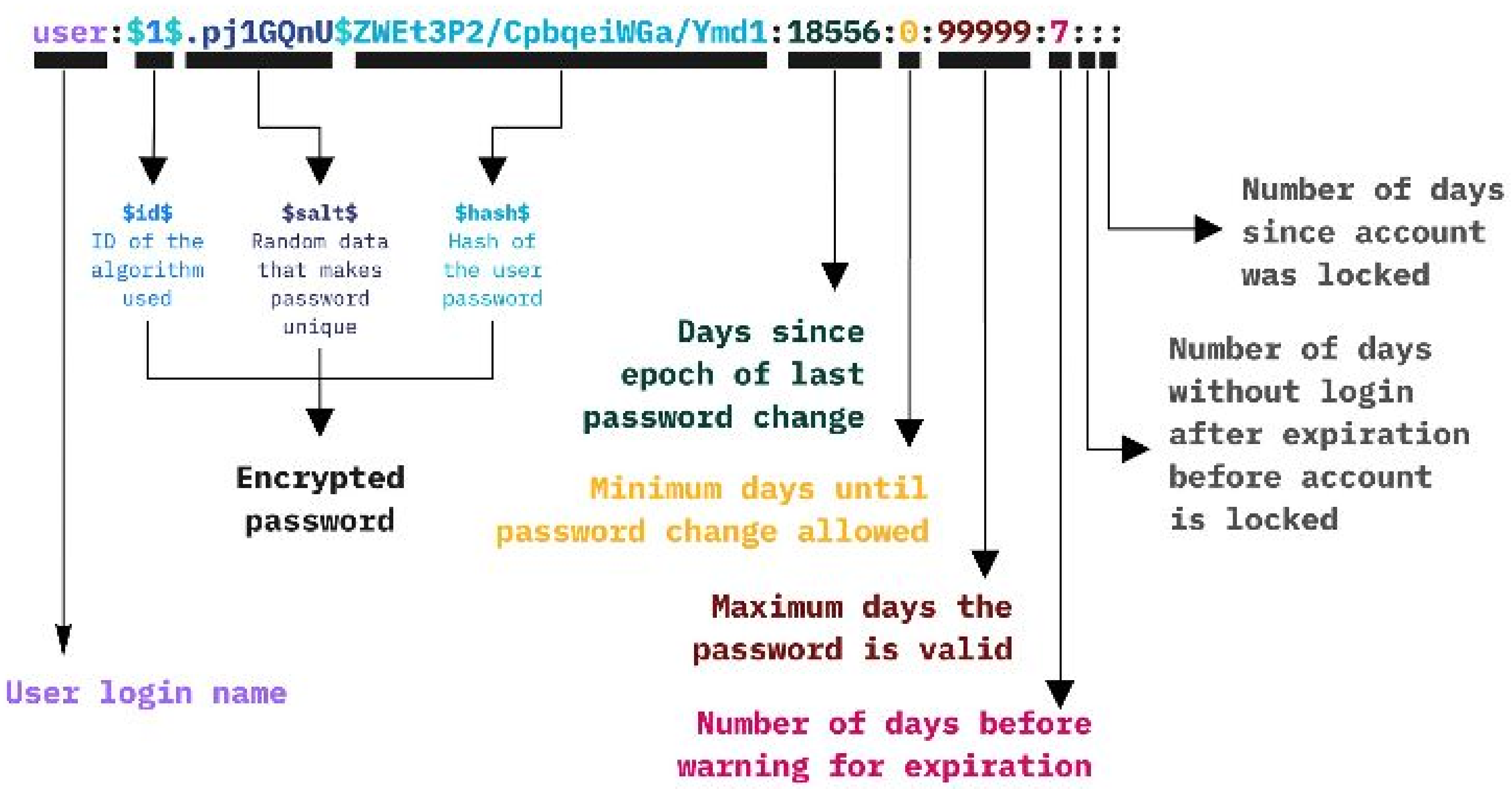
Preventing Dictionary Attacks

- Public salt:
  - When setting password, pick a random n-bit salt S
  - When verifying pw for A, test if $\mathbf{H(pw, S_A) = h_A}$

id	S	h
Alice	S_A	$H(pw_A , S_A)$
Bob	S_B	$H(pw_B , S_B)$
…	…	…

- Recommended salt length, $n = 64$ bits
  - Pre-hashing dictionary does not help

/etc/shadow file

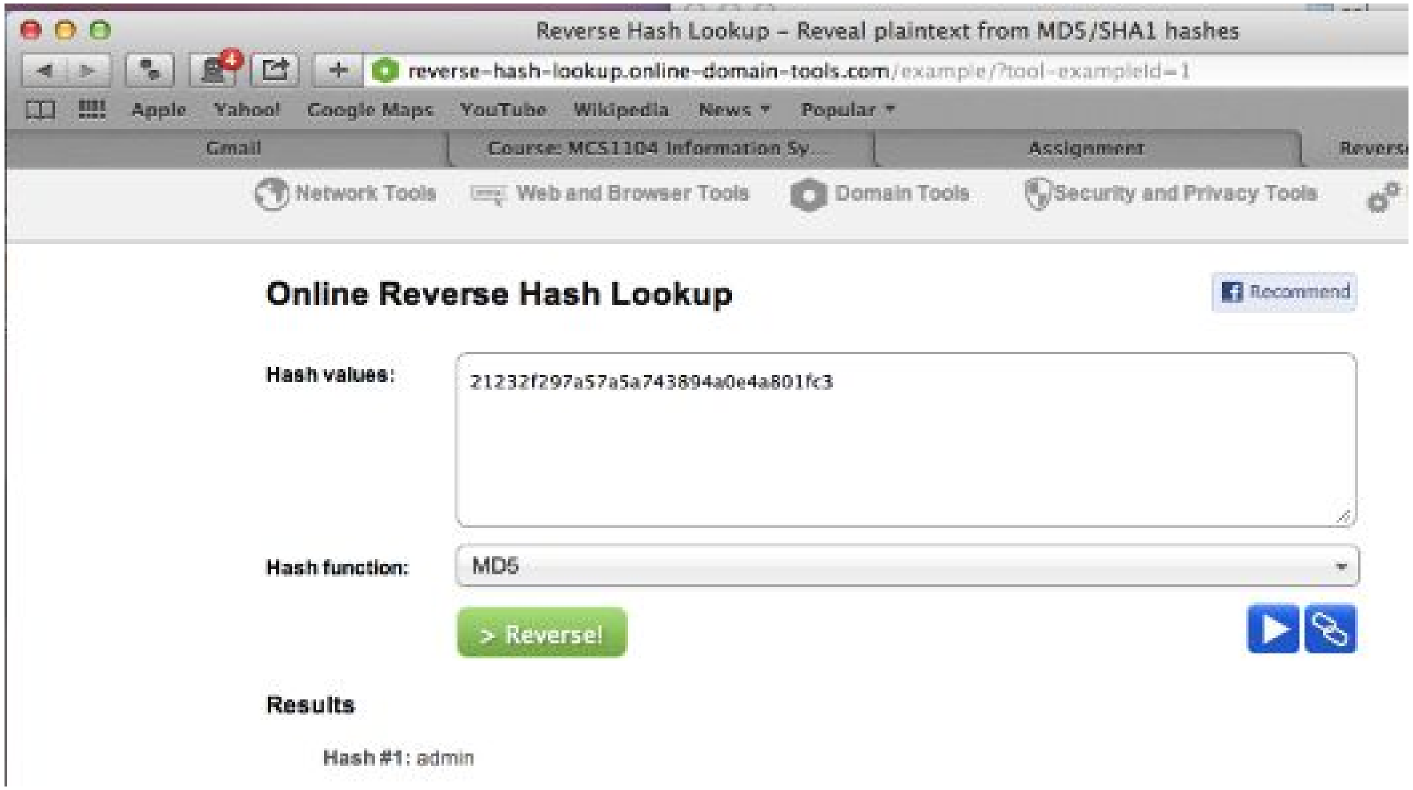


user:$1$.pj1GQnU$ZWEt3P2/CpbqeiWGa/Ymd1:18556:0:99999:7:::
id
ID of the algorithm used
$salt$
Random data that makes password unique
$hash$
Hash of the user password
Encrypted password
User login name
Days since epoch of last password change
Minimum days until password change allowed
Maximum days the password is valid
Number of days before warning for expiration
Number of days without login after expiration before account is locked
Number of days since account was locked

Password Hashing Competition (PHC)

- In July 2015 PHC announced **Argon2** as a winner and gave special recognition to four of the finalists:
    - **Catena**, for its agile framework approach and side-channel resistance (Catena-v5.tar.gz)
    - **Lyra2**, for its elegant sponge-based design, and alternative approach to side-channel resistance (Lyra2-v3.tar.gz)
    - **Makwa**, for its unique delegation feature and its factoring-based security (Makwa-v1.tar.gz)
    - **yescrypt**, for its rich feature set and easy upgrade path from scrypt (yescrypt-v2.tar.gz)

SHA1 Reverse Lookup


Reverse Hash Lookup – Reveal plaintext from MD5/SHA1 hashes
reverse-hash-lookup.online-domain-tools.com/example/?tool-exampleId=1
Apple Yahoo! Google Maps YouTube Wikipedia News Popular
Gmail
Course: MCS1104 Information Sy...
Assignment
Network Tools
Web and Browser Tools
Domain Tools
Security and Privacy Tools
Recommend
> Reverse!

Online Reverse Hash Lookup

Hash values: 21232f297a57a5a743894a0e4a801fc3

Hash function: MD5

Results

Hash #1: admin

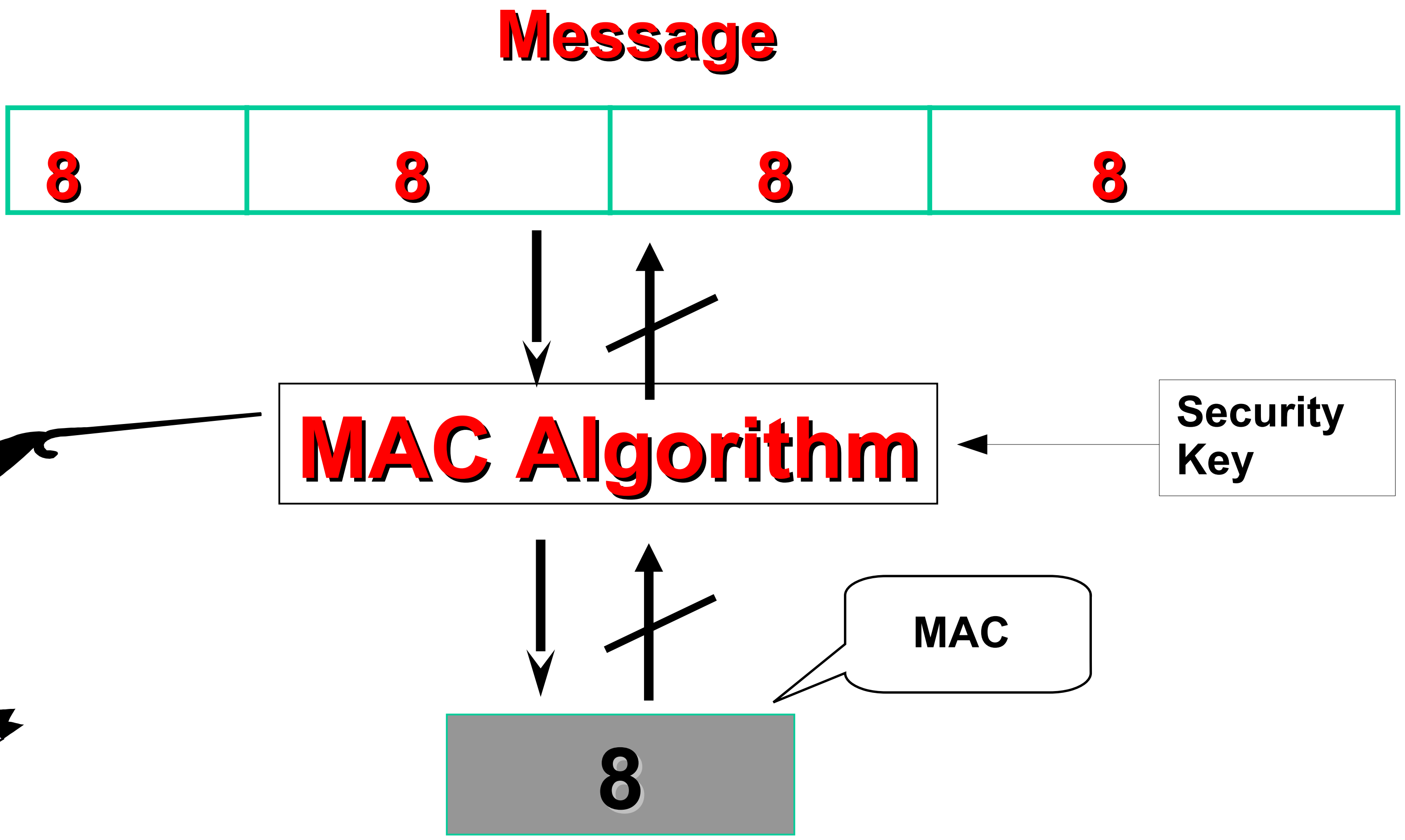
reverse-hash-lookup.online-domain-tools.com

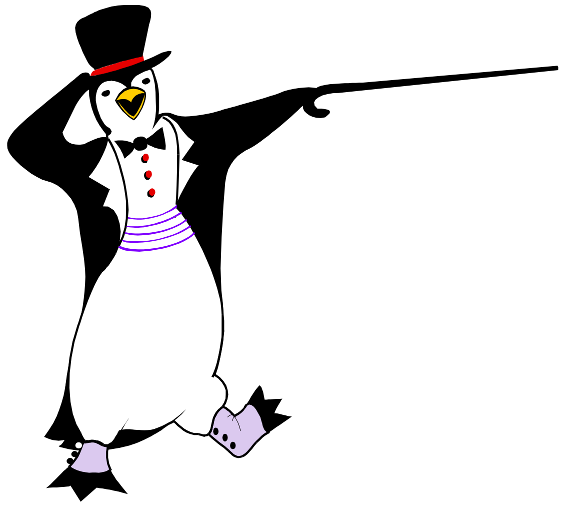
```
echo -n 'kasun'| md5
```

Message Authentication Code (MAC)

- Generated by an algorithm that creates a small fixed-sized block depending on both message and some key
- need not be reversible
- Receiver performs same computation on message and checks if it matches the MAC
- Provides assurance that message is unaltered and comes from sender

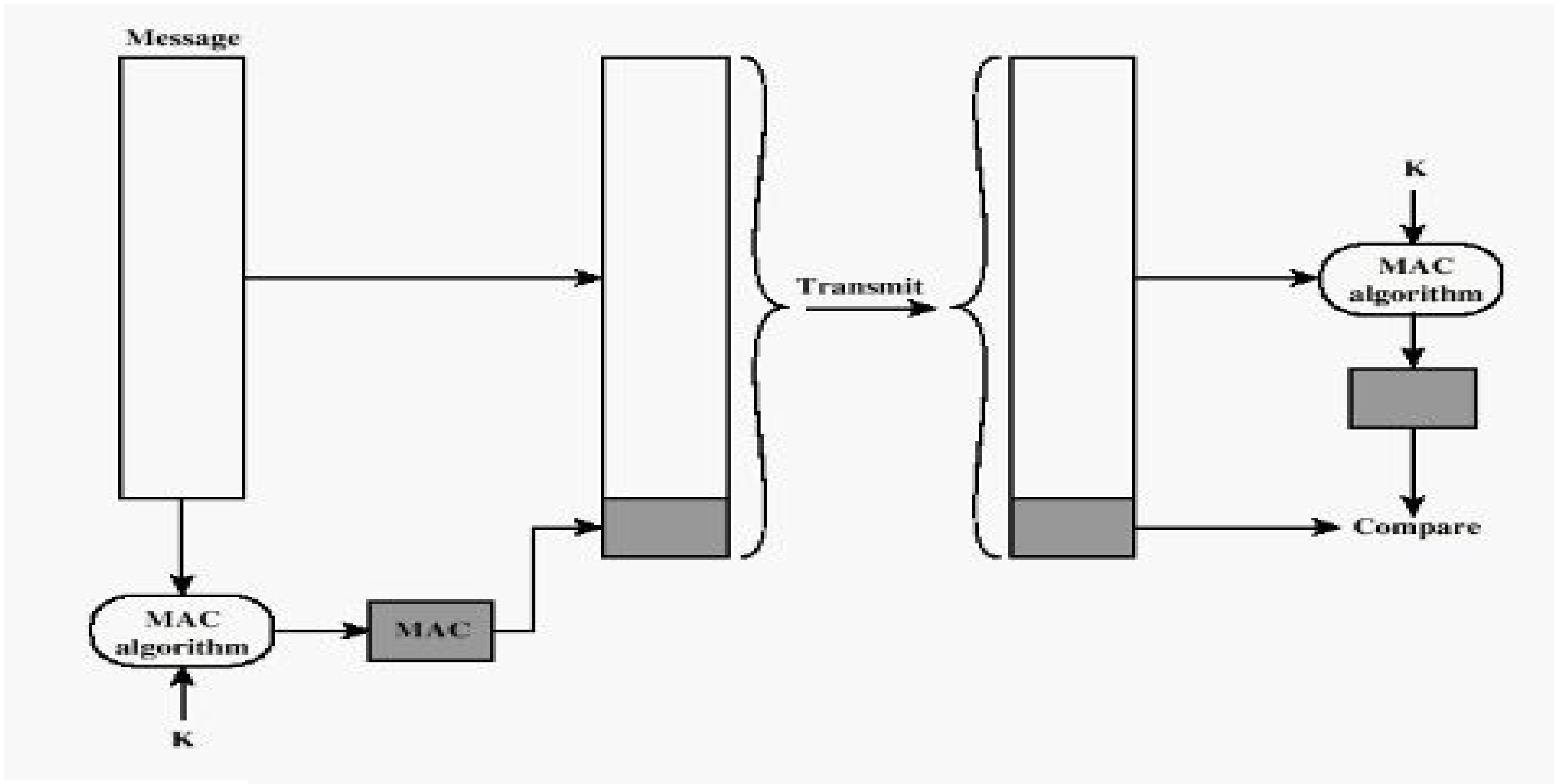
Message Authentication Code (MAC)





Approaches to Message Authentication

- Authentication Using Conventional Encryption
  - Only the sender and receiver should share a key
- Message Authentication without Message Encryption
  - An authentication tag is generated and appended to each message
- Message Authentication Code
  - Calculate the MAC as a function of the message and the key. MAC = F(K, M)



Message Authentication Using a Message Authentication Code (MAC)

Keyed Hash Functions (HMAC)

- Create a MAC using a hash function rather than a block cipher
  - because hash functions are generally faster
  - not limited by export controls unlike block ciphers
  - Hash includes a key along with the message
- Original proposal:

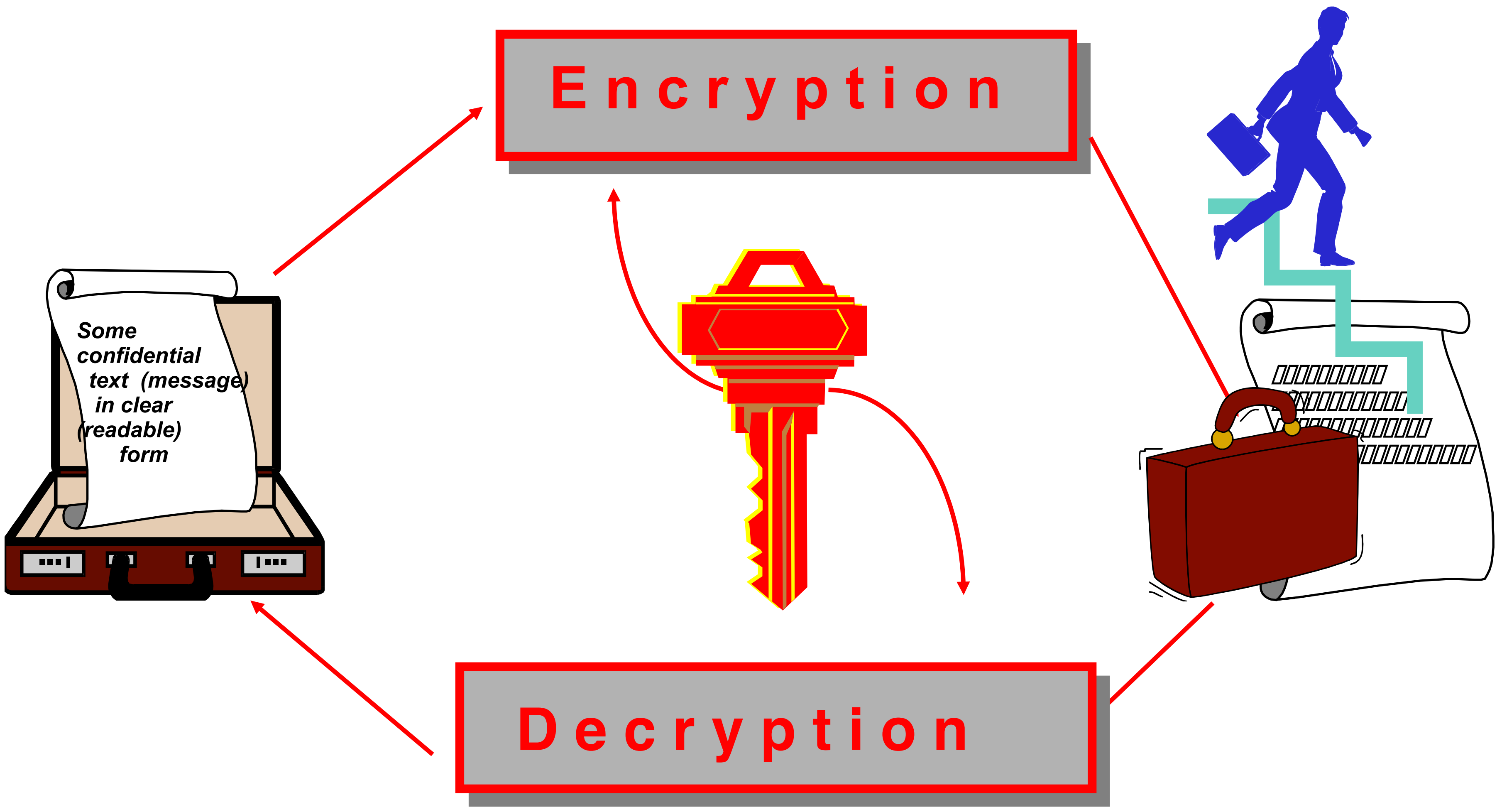
  $KeyedHash = Hash(Key|Message)$
  - some weaknesses were found with this
- Eventually led to development of HMAC

The classic cryptography

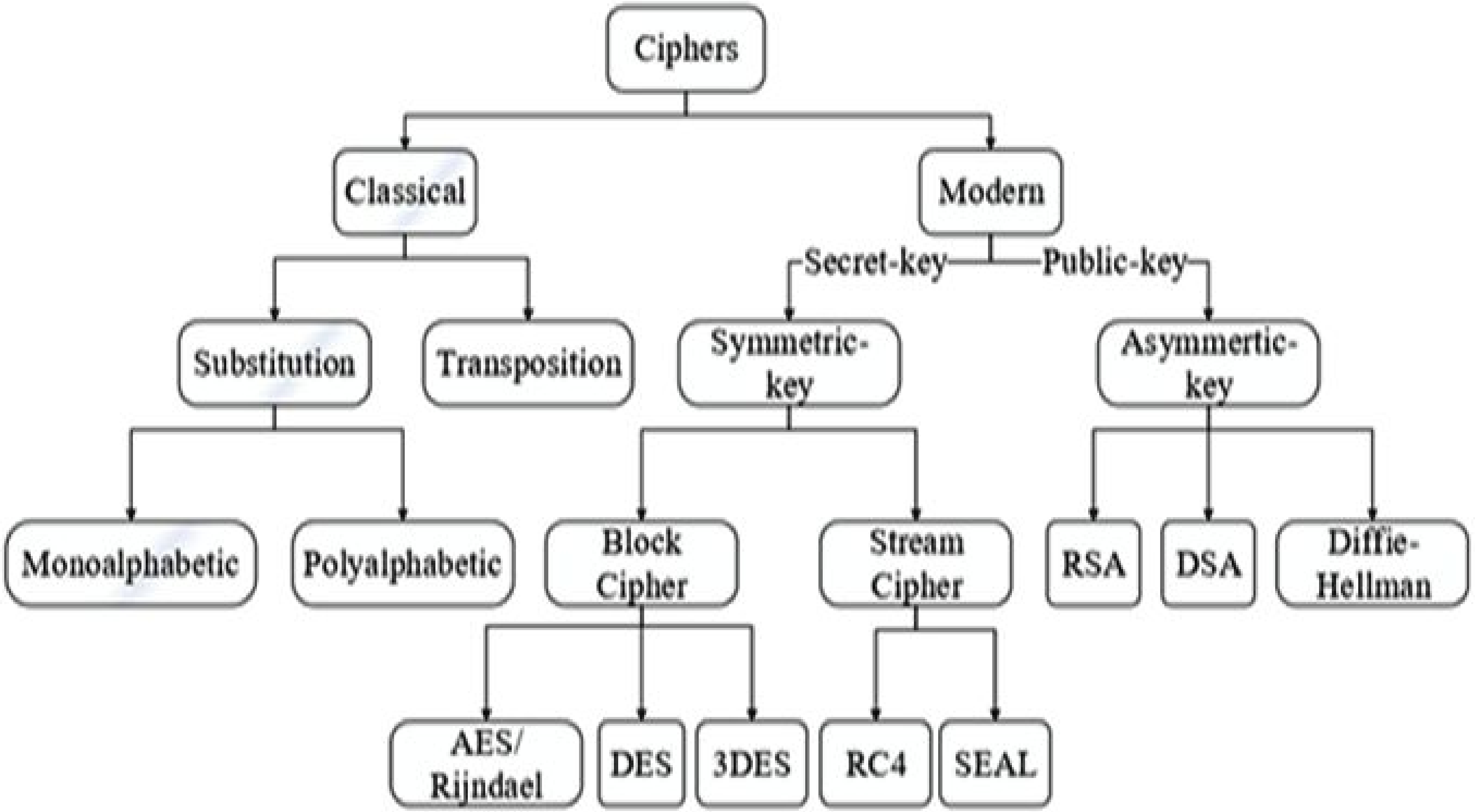
- **Encryption algorithm and related key are kept secret.**
- **Breaking the system is hard due to large numbers of possible keys.**
- **For example: for a key 128 bits long**
- **there are** keys to check $2^{128} \approx 10^{38}$ using brute force.

The fundamental difficulty is key distribution to parties who want to exchange messages.

Symmetric key Cryptograms



Ciphers

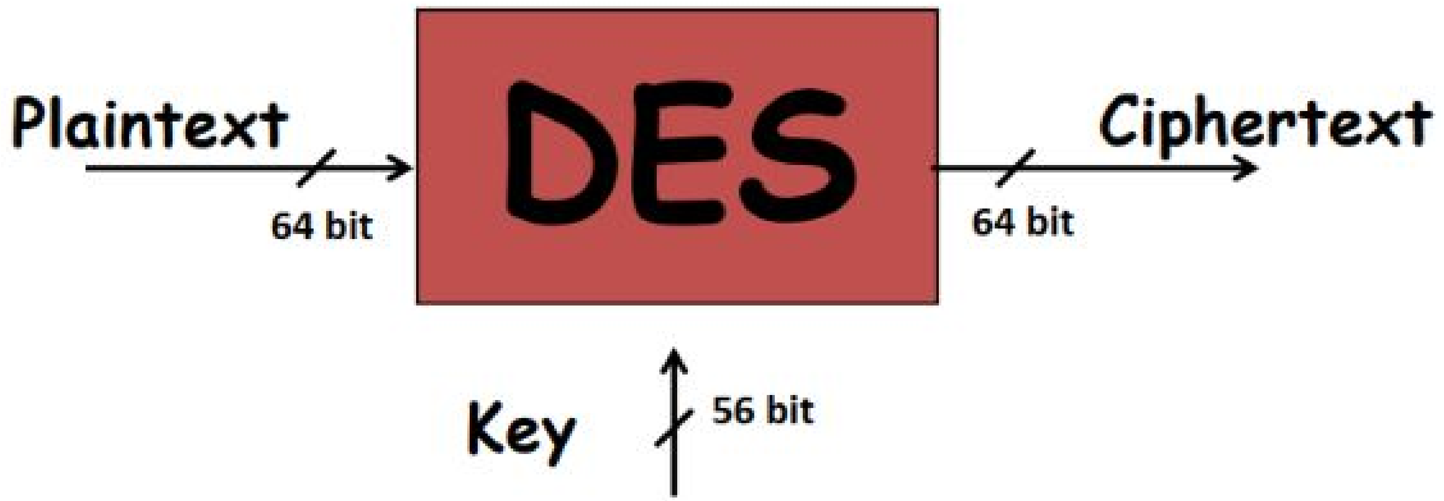

Ciphers
Classical
Modern
Secret-key
Public-key
Substitution
Transposition
Symmetric-key
Asymmertic-key
Monoalphabetic
Polyalphabetic
Block Cipher
Stream Cipher
RSA
DSA
Diffie-Hellman
AES/ Rijndael
DES
3DES
RC4
SEAL

Symmetric Key / Private Key Cryptosystem

- Uses a single Private Key shared between users
- Strengths
  - Speed/ Efficient Algorithms – much quicker than Asymmetric
  - Hard to break when using a large Key Size
  - Ideal for bulk encryption / decryption
- Weaknesses
  - Poor Key Distribution (must be done out of band – ie phone, mail, etc)
  - Poor Key Management / Scalability (each user needs a unique key)
  - Cannot provide authenticity or non-repudiation – only confidentiality

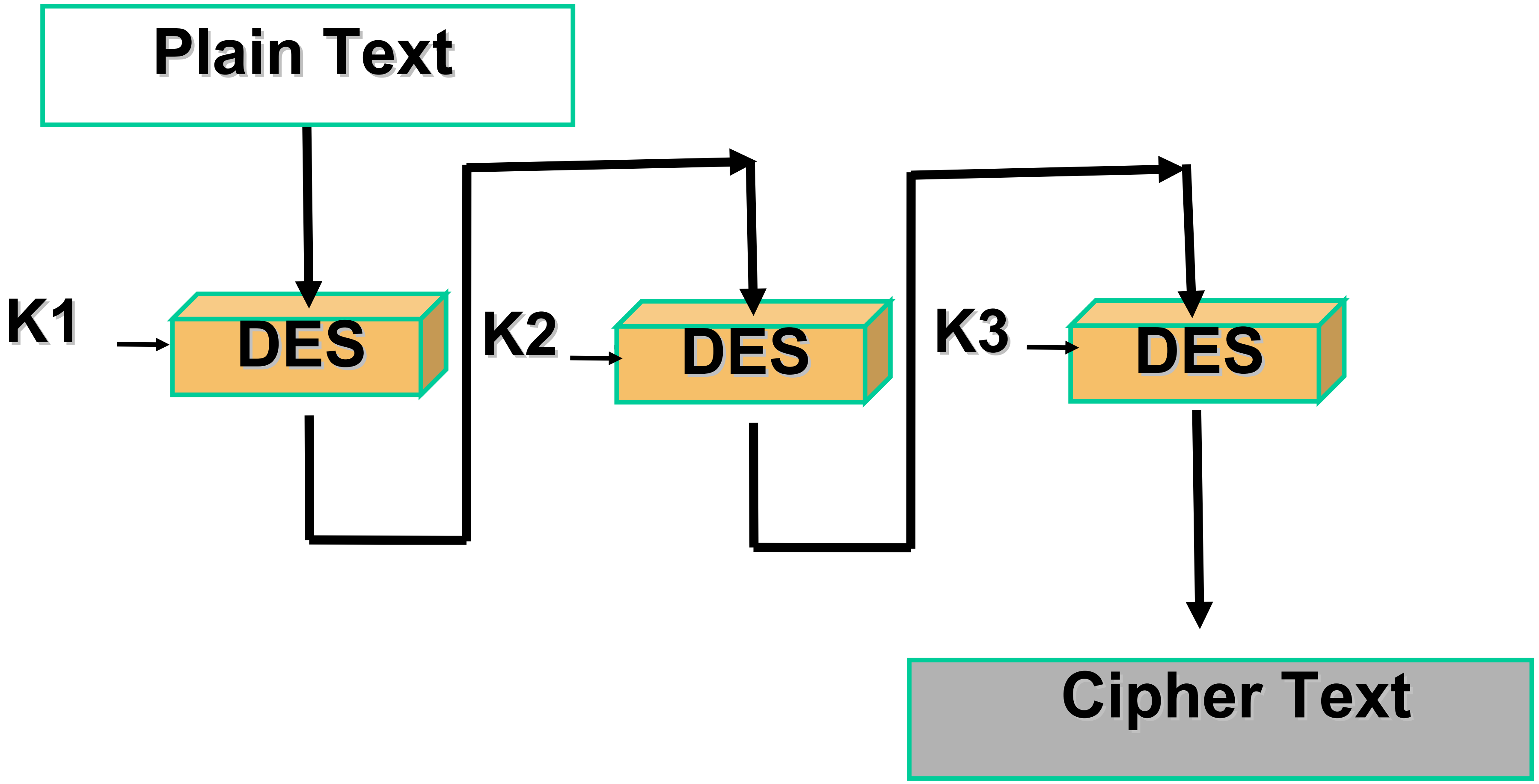
DES Features

- Features:
  - Block size = 64 bits
  - Key size = 56 bits (in reality, 64 bits, but 8 are used as parity-check bits for error control, see next slide)
  - Number of rounds = 16
  - 16 intermediary keys, each 48 bits

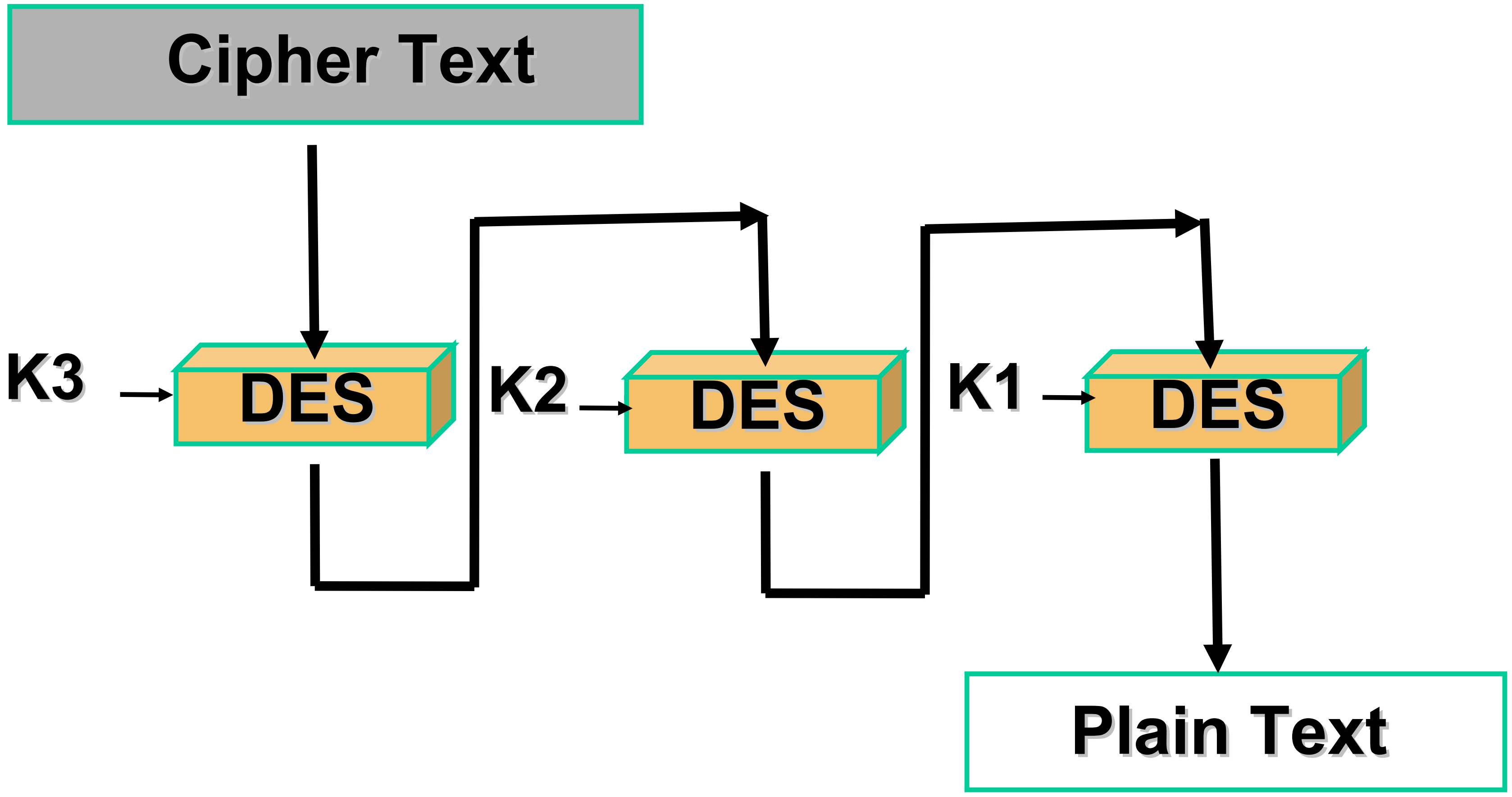


8

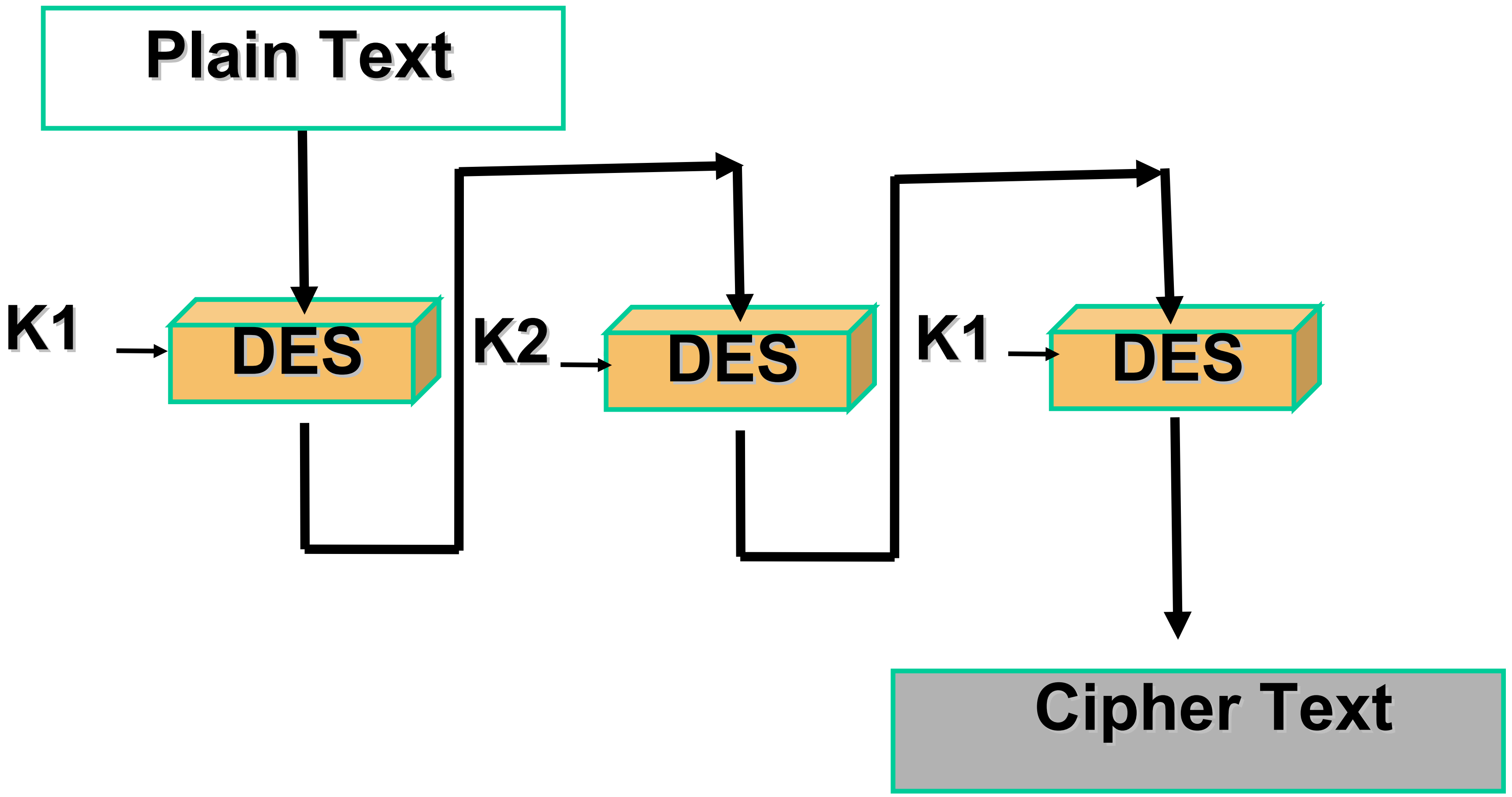
Triple DES - Encryption



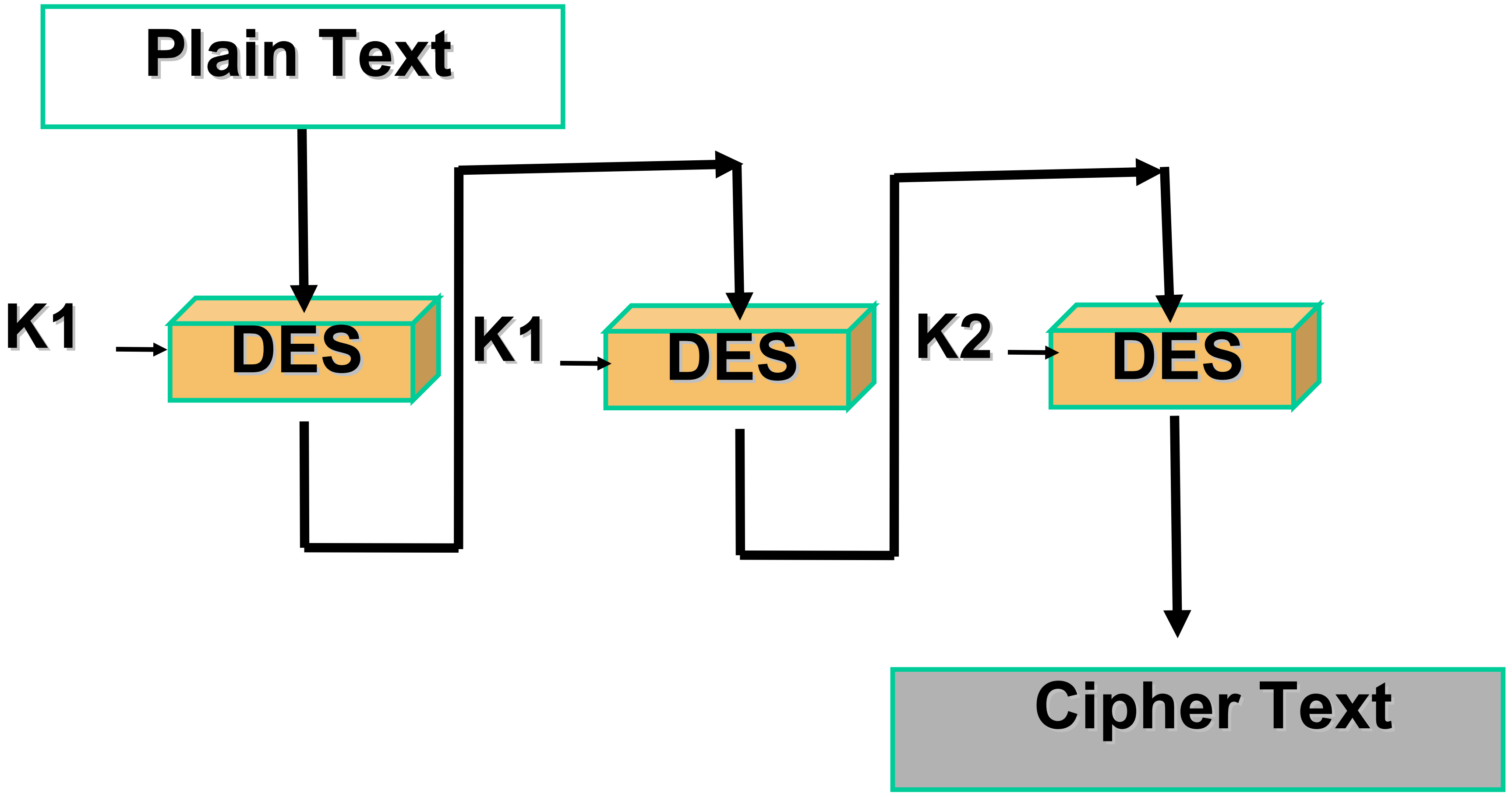
Triple DES - Decryption



Triple DES with Two Keys



Triple DES Backward Compatibility



DES- AES

- Clearly, a replacement for DES was needed
  - have theoretical attacks that can break it
  - have demonstrated exhaustive key search attacks
- Can use Triple-DES – but slow with small blocks
- NIST issued a call for ciphers in 1997
- 15 candidates accepted in June 1998
- 5 were short listed in August 1999
- Rijndael 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, 128/192/256-bit keys
- Stronger & faster than Triple-DES
- Active life of 20-30 years (+ archival use)
- Provide full specification & design details
- Both C & Java implementations
- NIST has released all submissions & unclassified analyses

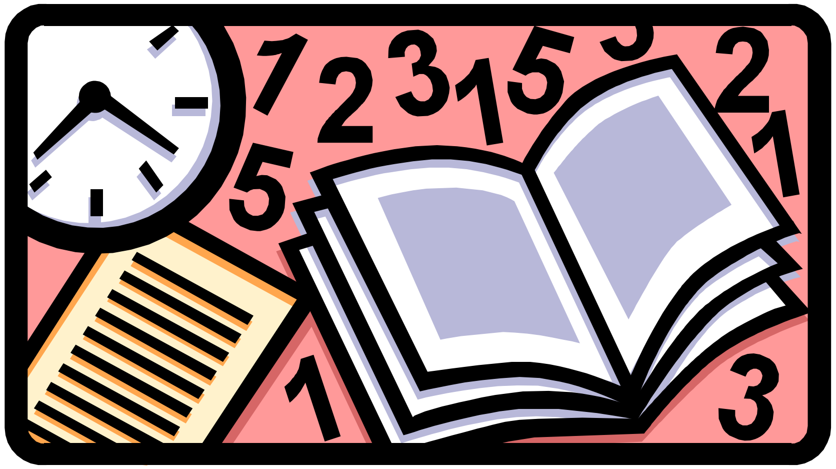
AES Shortlist

- After testing and evaluation, shortlist in August 1999:
  - MARS (IBM) - complex, fast, high security margin
  - RC6 (USA) - v. simple, v. fast, low security margin
  - Rijndael (Belgium) - clean, fast, good security margin
  - Serpent (Euro) - slow, clean, v. high security margin
  - Twofish (USA) - complex, v. fast, high security margin

- Then subject to further analysis & comment
- Saw contrast between algorithms with
  - few complex rounds verses many simple rounds
  - which refined existing ciphers verses new proposals

Advance Encryption Standard (AES)

- In 2001, National Institute of Standards and Technology (NIST) issued AES known as FIPS 197
- AES is based on Rijndael proposed by Joan Daemen, Vincent Rijmen from Belgium



Advance Encryption Standard (AES)

- AES has block length 128
- Supported key lengths are 128, 192 and 256
- AES requires 10 rounds of processing
- Key is expanded into 10 individual keys
- Decryption algorithm uses the expanded keys in reverse order
- Decryption algorithm is not identical to the encryption algorithm

Block Ciphers- Modes of Operation

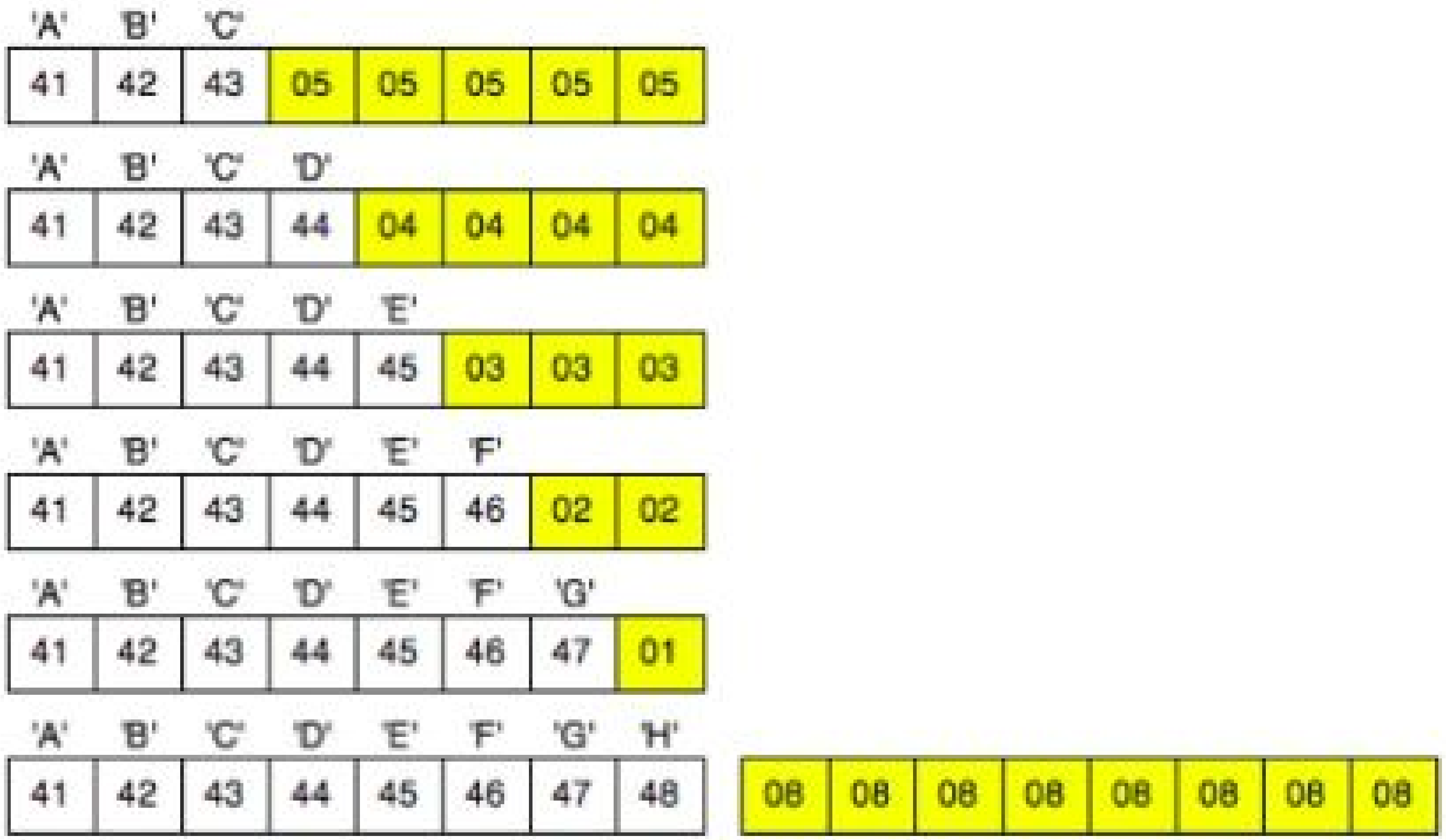
- Block ciphers encrypt fixed size blocks
  - E.g. DES encrypts 64-bit blocks, with 56-bit key
- Given that one needs to encrypt arbitrary amount of information, how do we use in practice,
  - Four modes were defined for DES in ANSI standard
  - **ANSI X3.106-1983 Modes of Use**
  - Subsequently now have 5 for DES and AES

PKCS5 Padding Scheme

- Assume block cipher is 64-bits
- Any message not a multiple of 8 bytes is padded
- Valid pad:
- 1 byte needed: 0x1
- 2 bytes needed: 0x2 0x2
- 3 bytes needed: 0x3 0x3 0x3
- ....
- No padding: 0x8 0x8 0x8 0x8 0x8 0x8 0x8 0x8

  (If the length of the original data is an integer multiple of the block size B, then an extra block of bytes with value B is added.)

PKCS5 Padding Scheme



'A'	'B'	'C'													
41	42	43	05	05	05	05	05								
'A'	'B'	'C'	'D'												
41	42	43	44	04	04	04	04								
'A'	'B'	'C'	'D'	'E'											
41	42	43	44	45	03	03	03								
'A'	'B'	'C'	'D'	'E'	'F'										
41	42	43	44	45	46	02	02								
'A'	'B'	'C'	'D'	'E'	'F'	'G'									
41	42	43	44	45	46	47	01								
'A'	'B'	'C'	'D'	'E'	'F'	'G'	'H'								
41	42	43	44	45	46	47	48	08	08	08	08	08	08	08	08

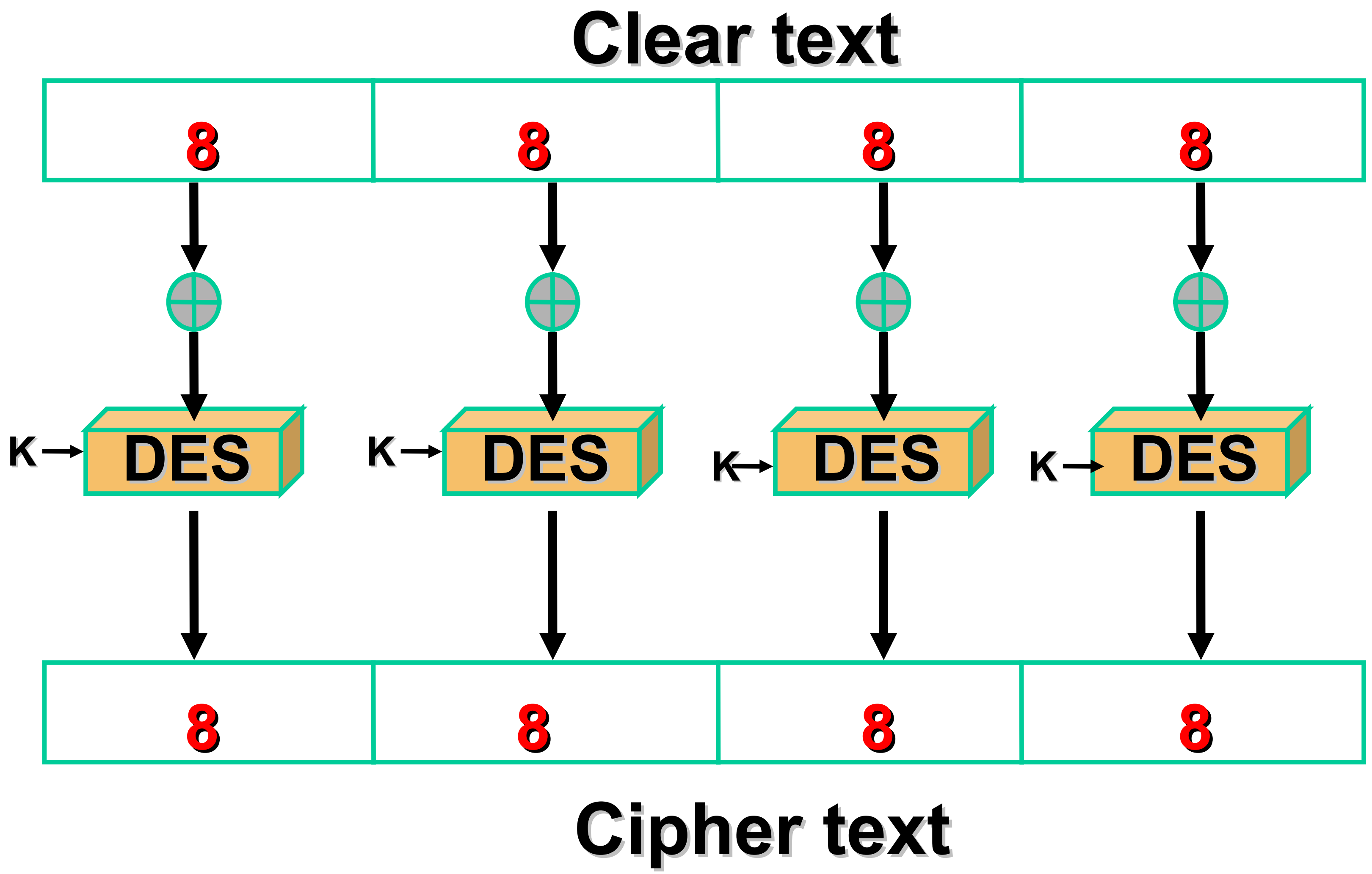
Electronic Codebook Book (ECB)

- Message is broken into independent blocks which are encrypted
- Each block is a value which is substituted, like a codebook, hence name
- Each block is encoded independently of the other blocks

$$C_i = DES_K(P_i)$$

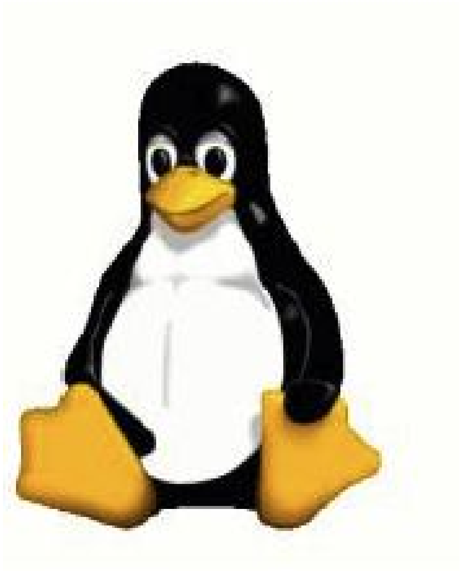
- Uses: secure transmission of single values

Electronic Code Book Mode (ECB)

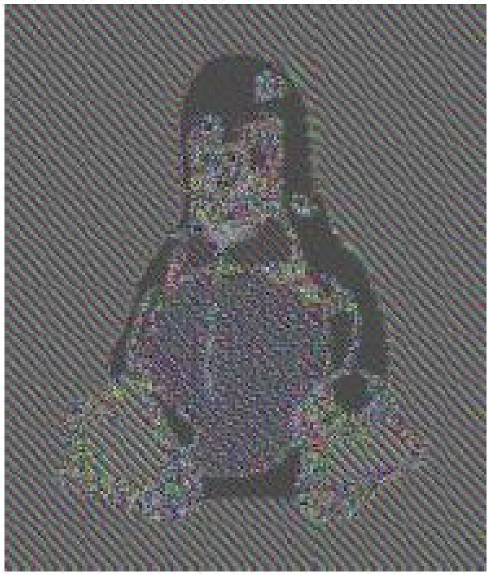


Advantages and Limitations of ECB

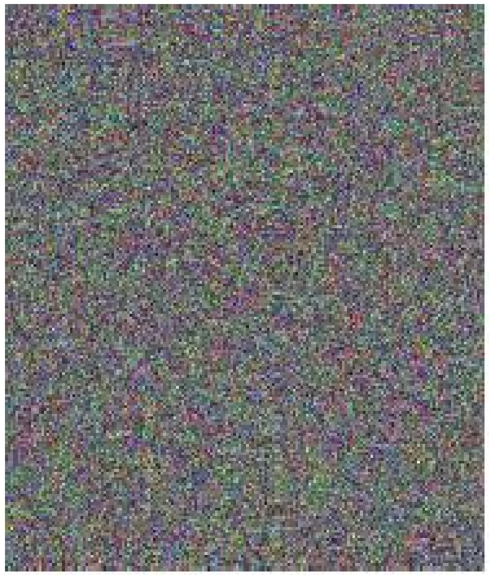
- Repetitions in message may show in ciphertext if aligned with message block particularly with data such graphics
- Messages that change very little
- Weakness due to encrypted message blocks being independent
- Main use is sending a few blocks of data



Original



Encrypted using ECB mode



Encrypted using other modes

Electronic codebook (ECB), Cipher block chaining (CBC), Cipher feedback (CFB), Output feedback (OFB)

Cipher Block Chaining (CBC)

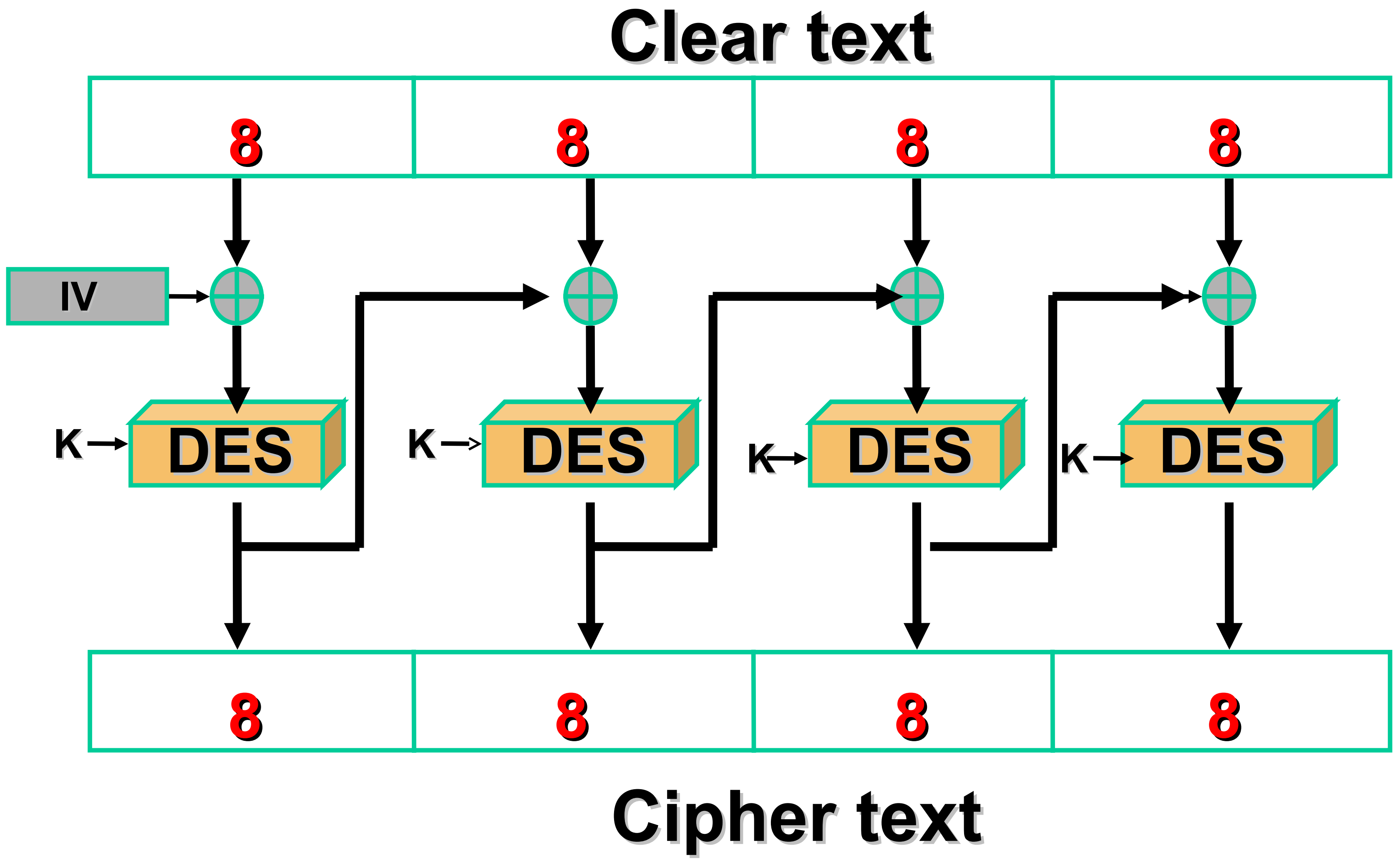
- Message is broken into blocks
- But these are linked together in the encryption operation
- Each previous cipher blocks is chained with current plaintext block, hence name
- Use Initial Vector (IV) to start process

$$C_i = DES_K(P_i \; XOR \; C_{i-1})$$

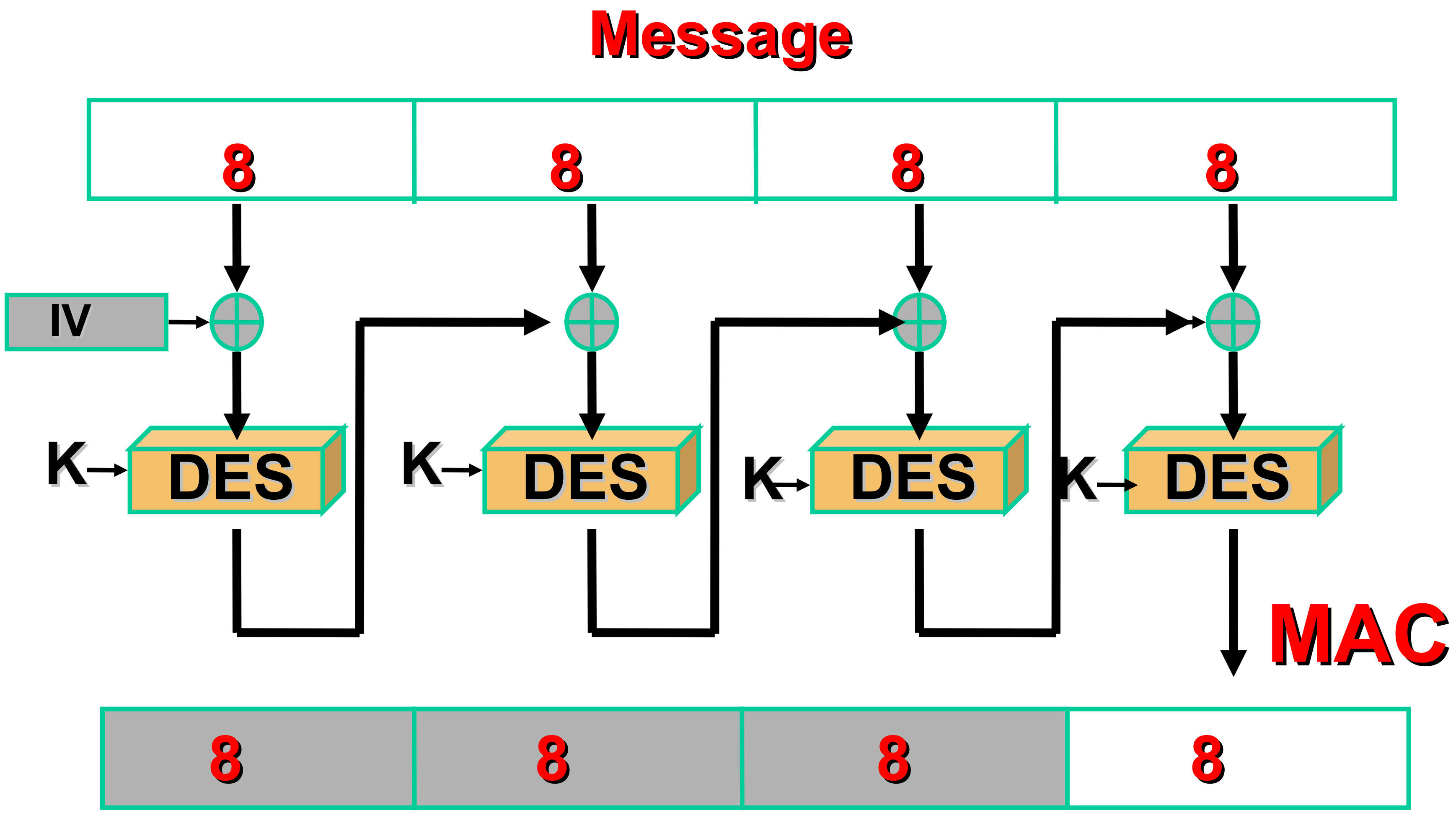
$$C_{-1} = IV$$

- Uses: bulk data encryption, authentication

Cipher Block Chaining Mode (CBC)



MAC based on CBC



Advantages and Limitations of CBC

- Each ciphertext block depends on **all** preceding message blocks thus a change in the message affects all ciphertext blocks after the change as well as the original block

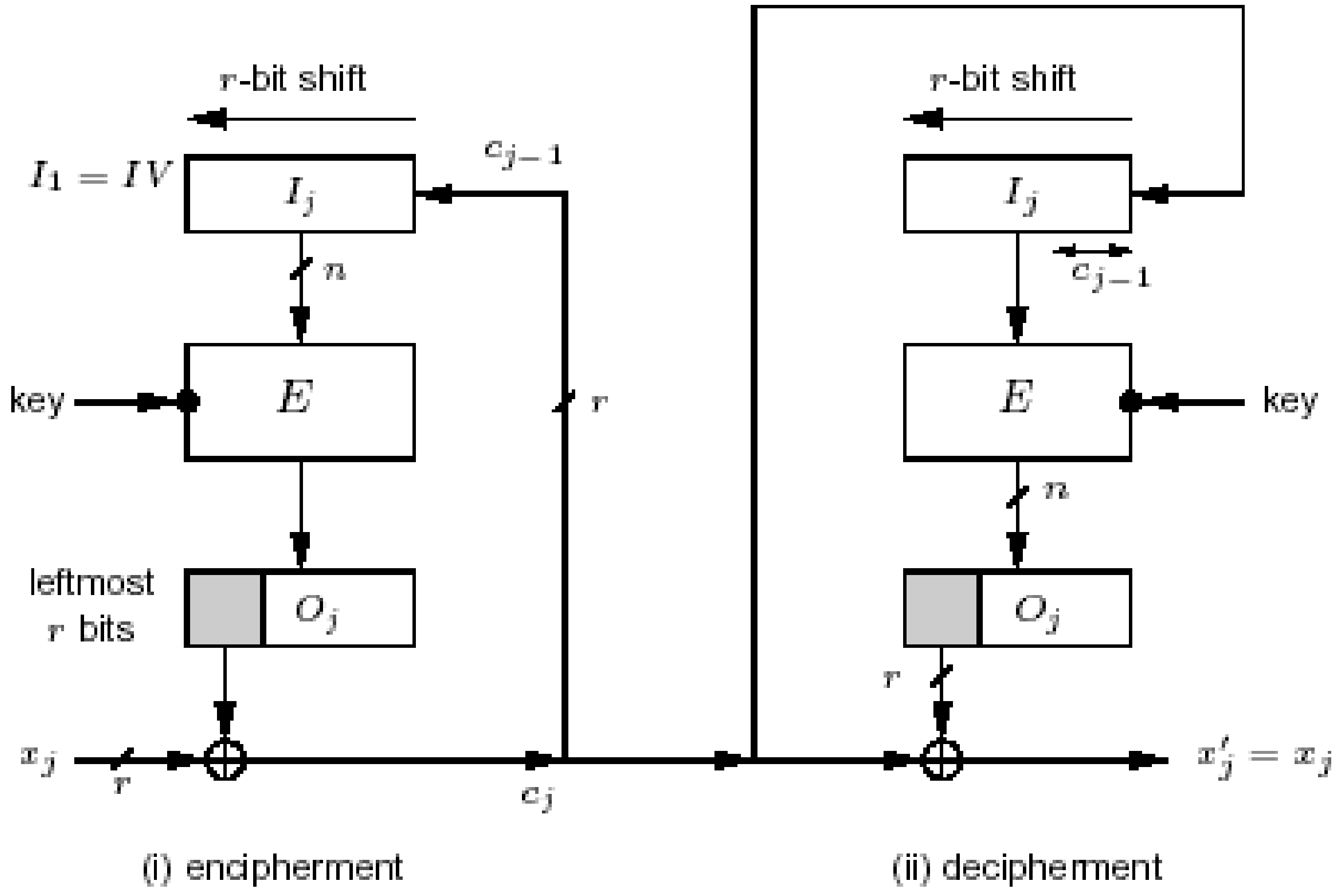
- Need **Initial Value** (IV) known to sender & receiver however if IV is sent in the clear, an attacker can change bits of the first block, and change IV to compensate hence either IV must be a fixed value or it must be sent encrypted in ECB mode before rest of message

- At end of message, handle possible last short block by padding either with known non-data value (e.g. nulls) or pad last block with count of pad size

Cipher feed back (CFB) mode

- A Stream Cipher where the Ciphertext is used as feedback into the Key generation source to develop the next Key Stream
- The Ciphertext generated by performing an XOR on the Plaintext with the Key Stream the same number of bits as the Plaintext
- Errors will propagate in this mode

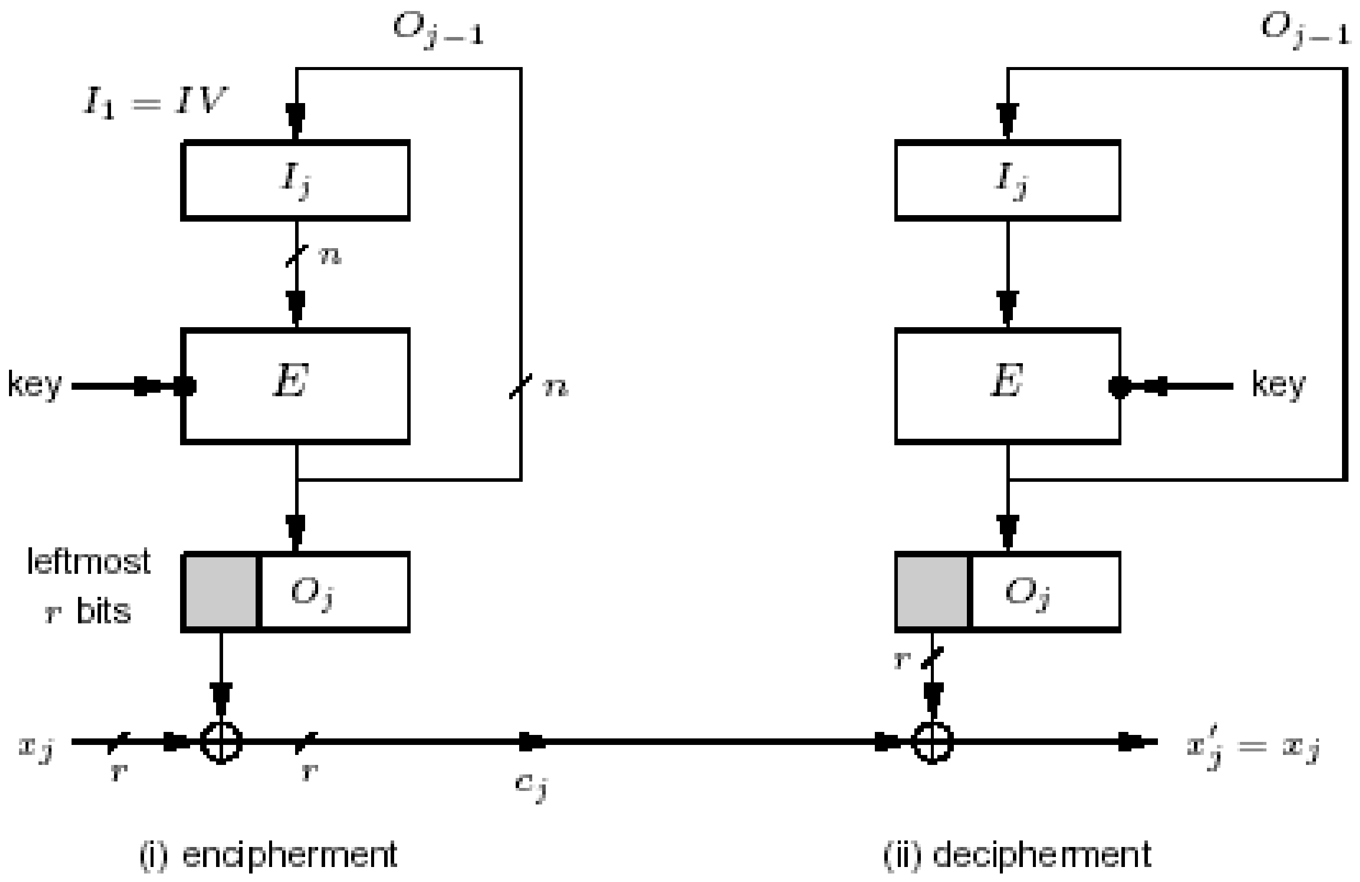
Cipher Feedback Mode (CFB)



Output Feed Back(OFB) mode

- A Stream Cipher that generates the Ciphertext Key by XORing the Plaintext with a Key Stream.
- Requires an Initialization Vector
- Feedback is used to generate the Key Stream – therefore the Key Stream will vary
- Errors will not propagate in this mode

Output Feedback Mode (OFB)



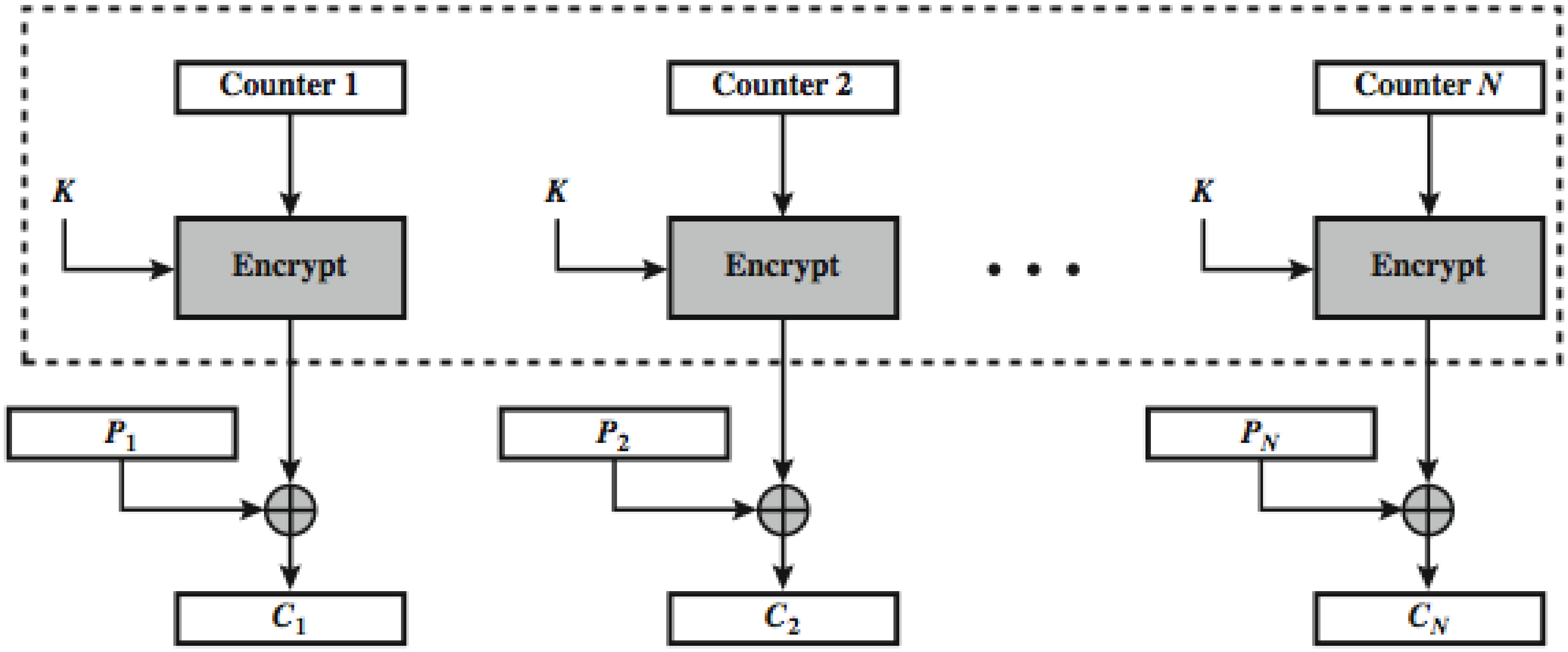
Counter (CTR)

a “new” mode, though proposed early on
similar to OFB but encrypts counter value rather than any feedback value

Oi = EK(i)
Ci = Pi XOR Oi

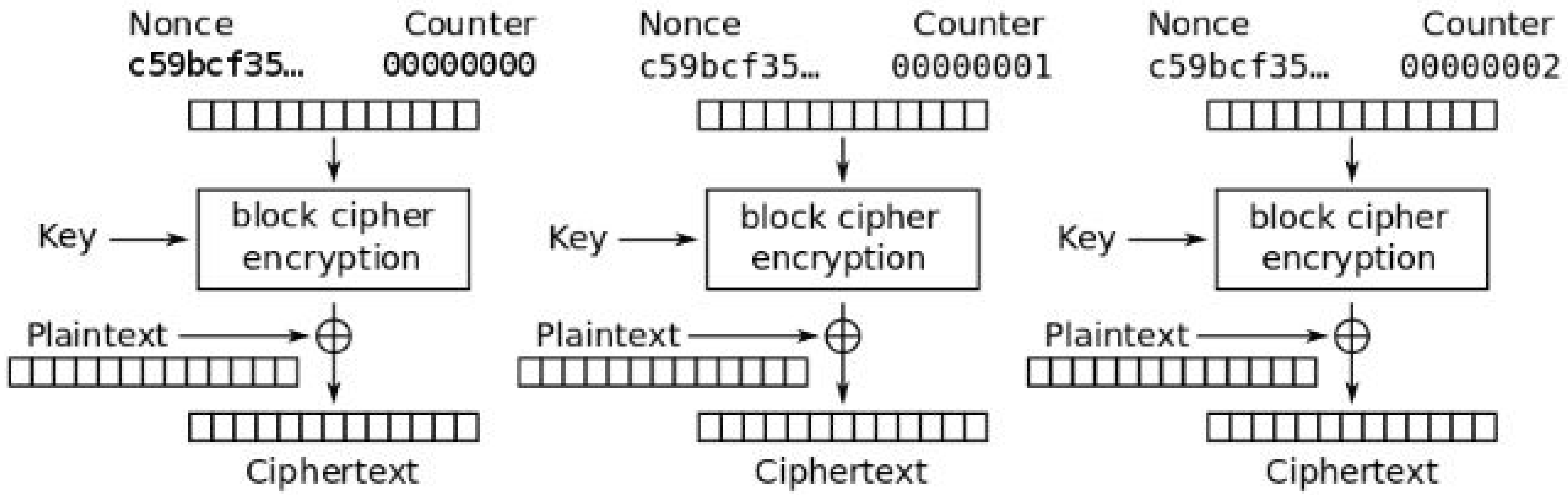
must have a different key & counter value for every plaintext block (never reused)
again
uses: high-speed network encryptions

CTR



(a) Encryption

CTR


Nonce
c59bcf35...
Counter
00000000
Key
block cipher encryption
Plaintext
Ciphertext
Nonce
c59bcf35...
Counter
00000001
Key
block cipher encryption
Plaintext
Ciphertext
Nonce
c59bcf35...
Counter
00000002
Key
block cipher encryption
Plaintext
Ciphertext

Advantages and Limitations of CTR

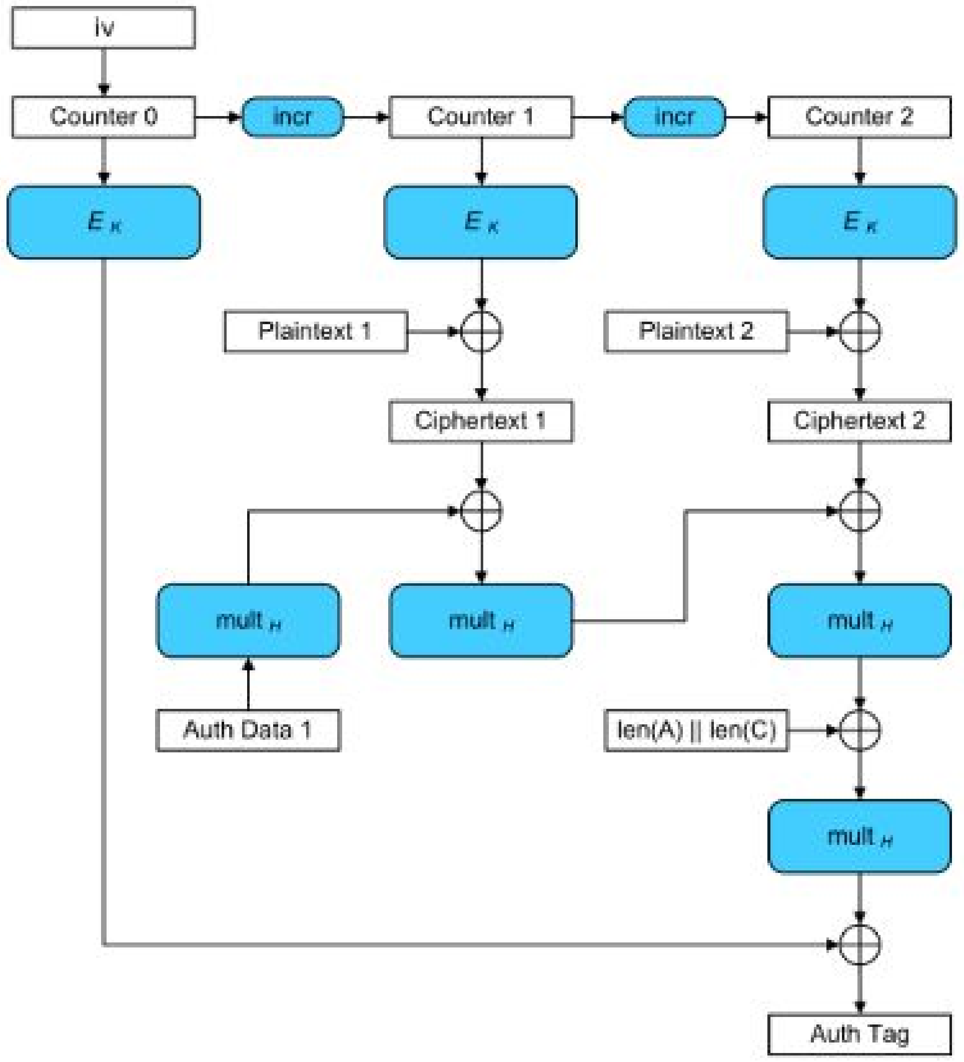
- can do parallel encryptions in h/w or s/w
- can preprocess in advance of need
- good for high speed links
- random access to encrypted data blocks
- provable security (good as other modes)
- but must ensure never reuse key/counter values, otherwise could break

GCM (Galois/Counter) Block Mode

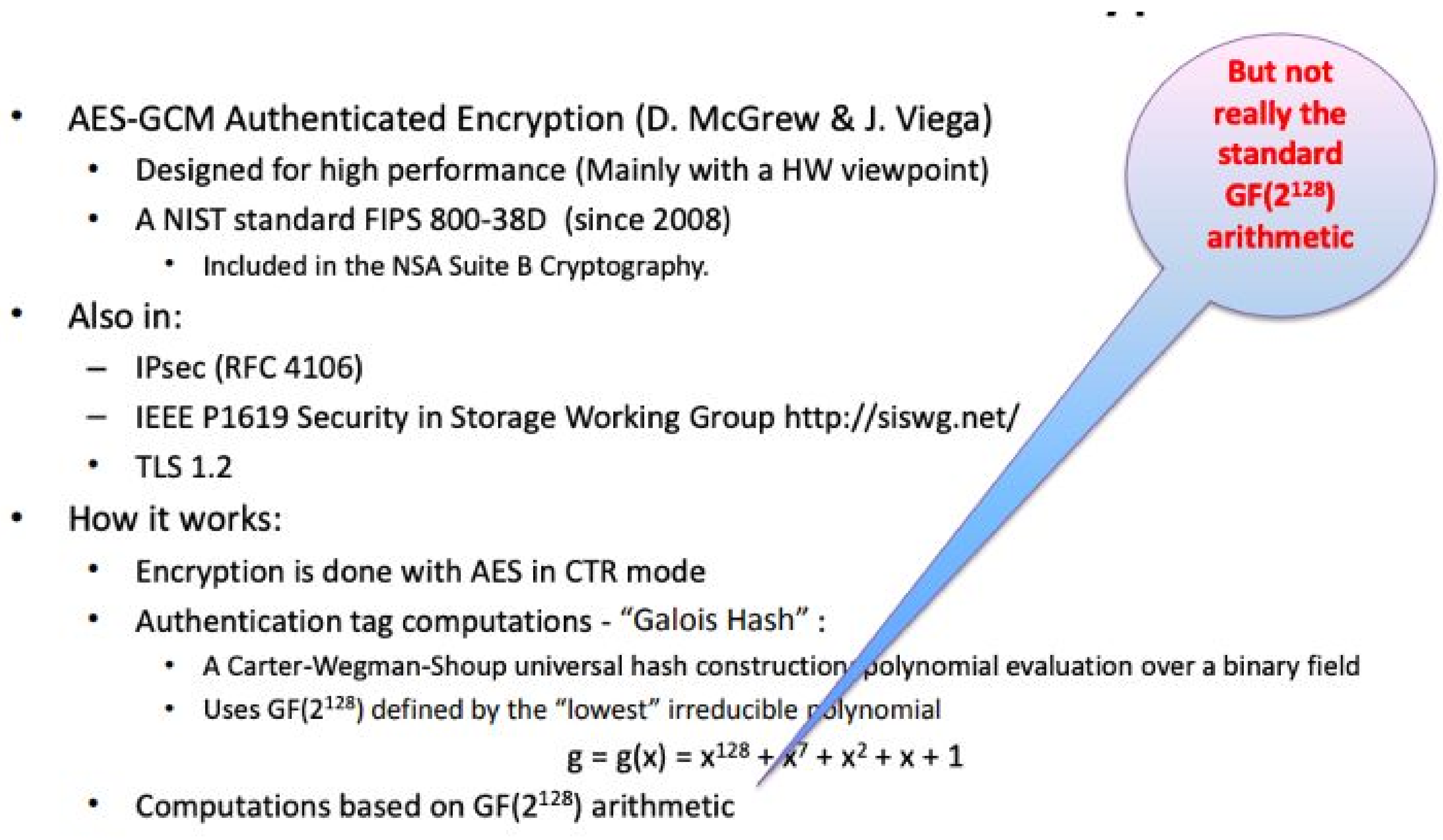
The GCM mode uses a counter, which is increased for each block and calculated a message authentication tag (MAC code) after each processed block.

The final authentication tag is calculated from the last block. Like all counter modes, GCM works as a stream cipher, and so it is essential that a different IV is used at the start for each stream that is encrypted.

The key-feature is the ease of parallel-computation of the Galois field multiplication used for authentication.



AES-GCM Authenticated Encryption



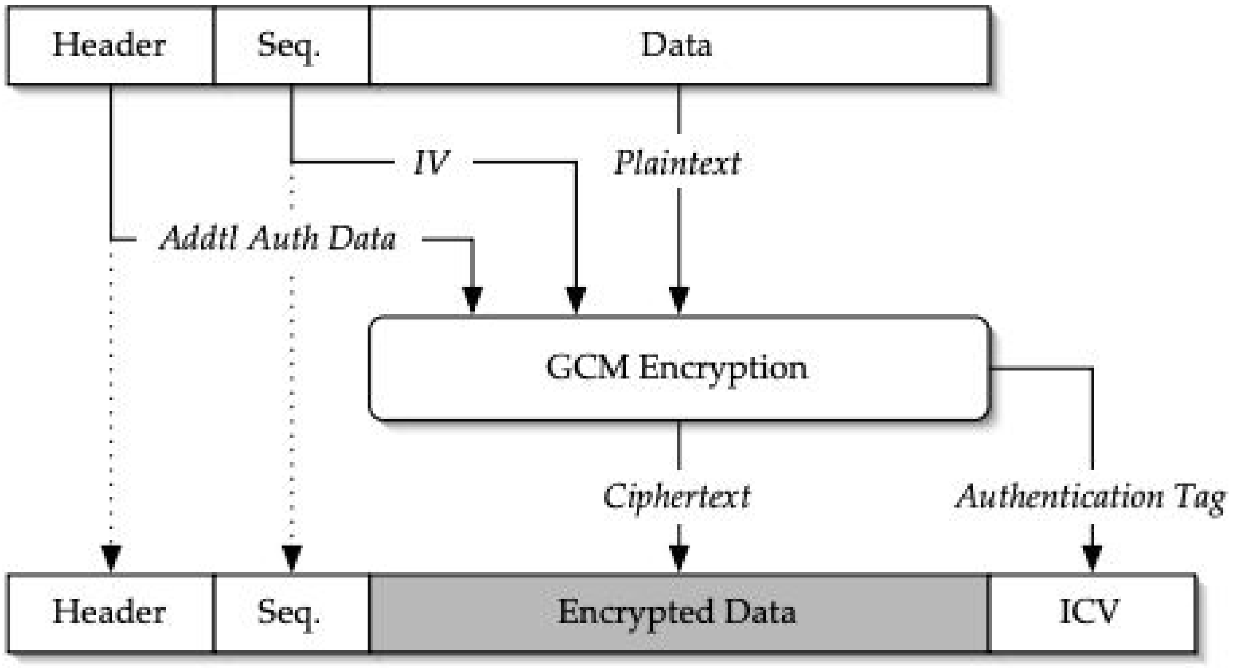
But not really the standard GF(2^{128}) arithmetic

- AES-GCM Authenticated Encryption (D. McGrew & J. Viega)
  - Designed for high performance (Mainly with a HW viewpoint)
  - A NIST standard FIPS 800-38D (since 2008)
    - Included in the NSA Suite B Cryptography.
- Also in:
  - IPsec (RFC 4106)
  - IEEE P1619 Security in Storage Working Group http://siswg.net/
  - TLS 1.2
- How it works:
  - Encryption is done with AES in CTR mode
  - Authentication tag computations - "Galois Hash" :
    - A Carter-Wegman-Shoup universal hash construction: polynomial evaluation over a binary field
    - Uses GF(2^{128}) defined by the "lowest" irreducible polynomial

$$g = g(x) = x^{128} + x^7 + x^2 + x + 1$$

  - Computations based on GF(2^{128}) arithmetic

AES- GCM



AES-GCM is the best performing Authenticated Encryption combination among the NIST standard options (esp. compared to using HMAC SHA-1)

OpenSSL

# encrypt file.txt to file.enc using 256-bit AES in CBC mode

```
>openssl enc -aes-256-cbc -in file.txt -out file.enc
```

# decrypt binary file.enc

```
>openssl enc -d -aes-256-cbc -in file.enc
```

# see the list under the 'Cipher commands' heading

```
>openssl -h
```

Advantages & Disadvantages

<u>**Advantages**</u>

Algorithms are fast

- ***Encryption & decryption are handled by same key***
- ***As long as the key remains secret, the system also provide authentication***

<u>**Disadvantages**</u>

Key is revealed, the interceptors can decrypt all encrypted information

- ***Key distribution problem***
- ***Number of keys increases with the square of the number of people exchanging secret information***

Discussion