SWE3009: : Internet Service and Computer Security
Lecture 0x05: Crypto and TLS

Hojoon Lee

Systems Security Lab @ SKKU

Crypto Concepts
(From Security Engineer and Programmer Perspectives)

Systems Security Lab

Objectives of this Lecture

- Crypto Basics
- Symmetric Key Cryptography
- Asymmetric Key Cryptography (Public Key)
- Hash Functions, MAC, HMAC
- Shared secret Generation (Diffie-Hellman Key Exchange)
- Final Objective: How all of the above work together in SSL/TLS

Systems Security Lab

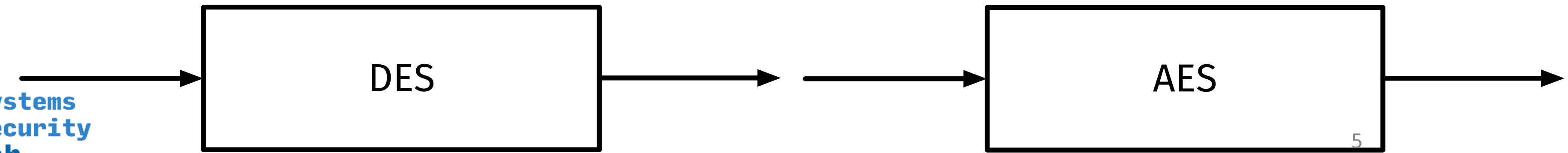
SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Disclaimer

- In this course, you are not required to understand <u>cryptography-side</u> of the things we learn
- We will focus on understanding the <u>big picture</u>
  - How today's secure communication and protocols are built using crypto
  - How crypto algorithms are applied for integrity, confidentiality etc..

Disclaimer

- Rationale: security engineers and programmers today use well-established crypto algorithms as building blocks
- In most cases, programmers must have a good understanding of What they do , but not necessarily How they work,

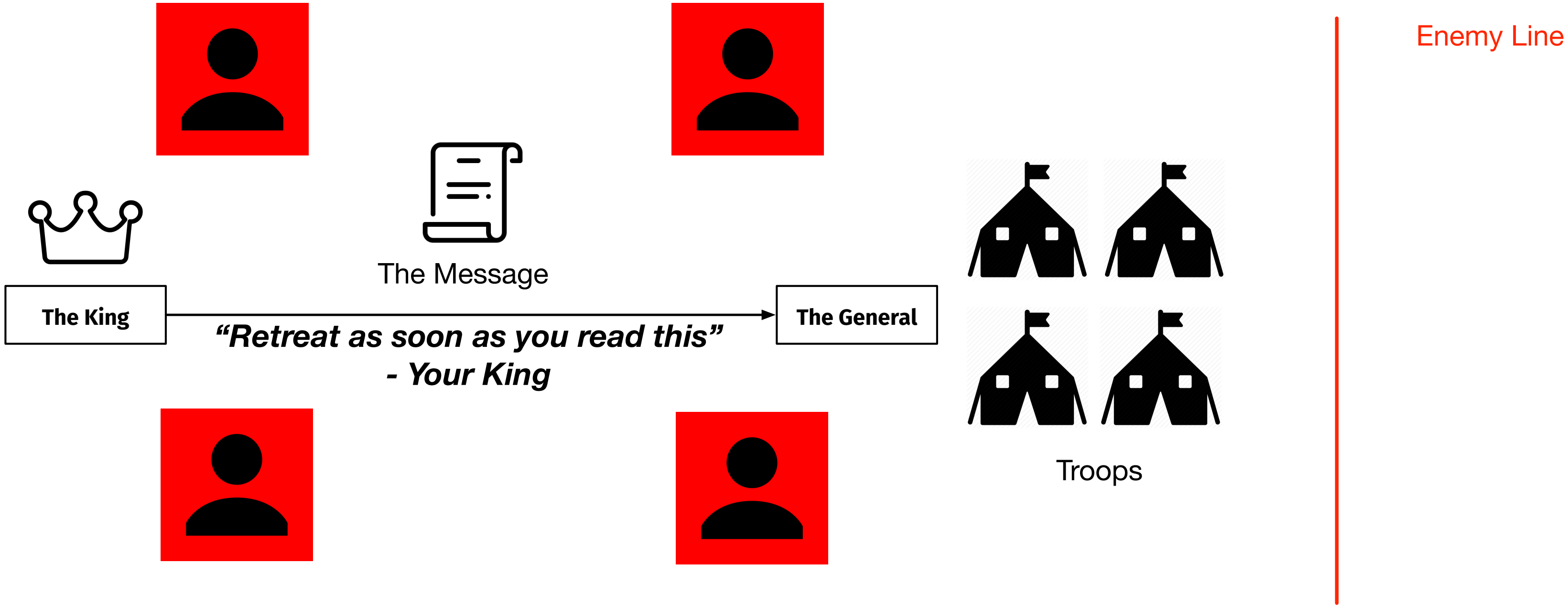


Crypto Overview

- The main objective of cryptography is to secure communication over insecure communication channels
- Security Goals
  - Confidentiality
  - Integrity
  - Authenticity

Crypto in History

Overview: Problem of thousands of years



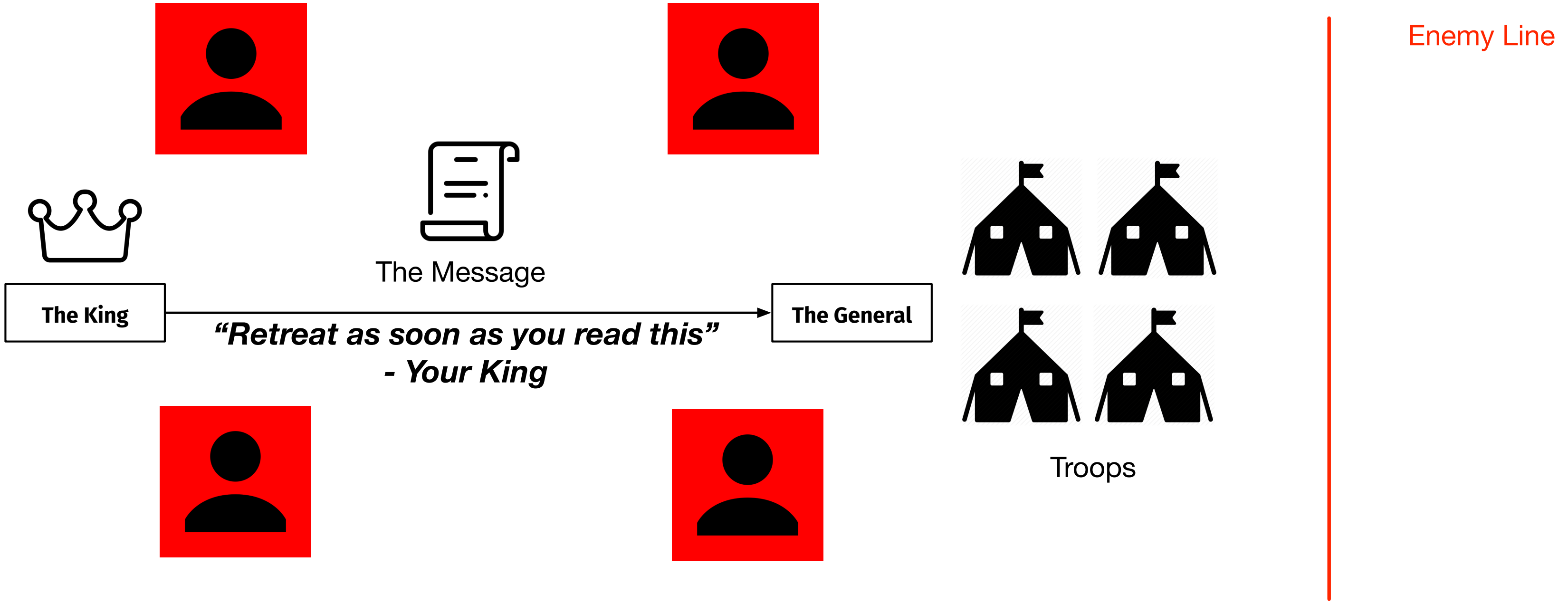
Confidentiality

- Even if our enemy captures the messenger and gets hold of the lettter,
- They must not be able to understand the message

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

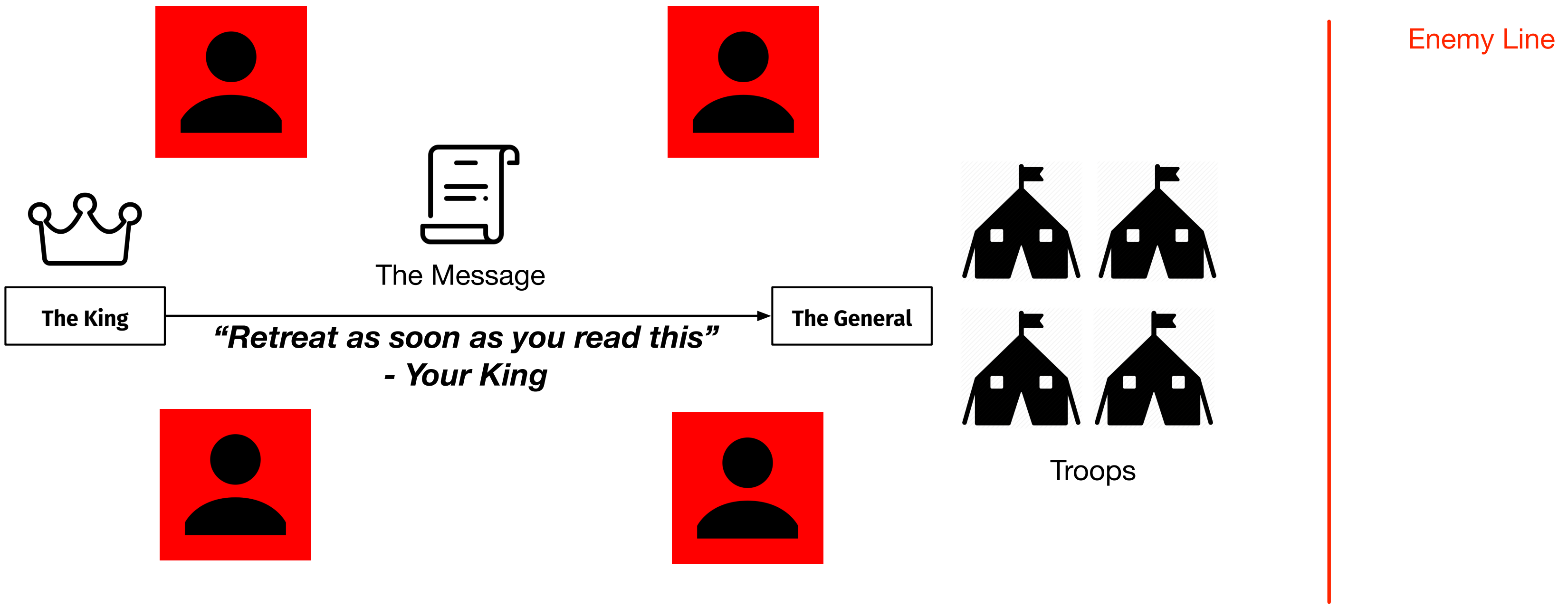
Overview: Problem of thousands of years



Integrity

- What if the message Had been stolen and already modified?
- What if the initial message was "You must hold the line until..."

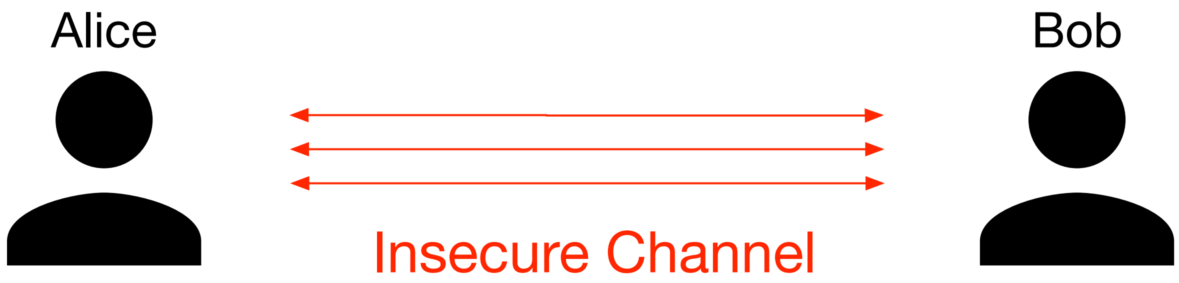
Overview: Problem of thousands of years



Authenticity

- How do we know if the message is even from our King?

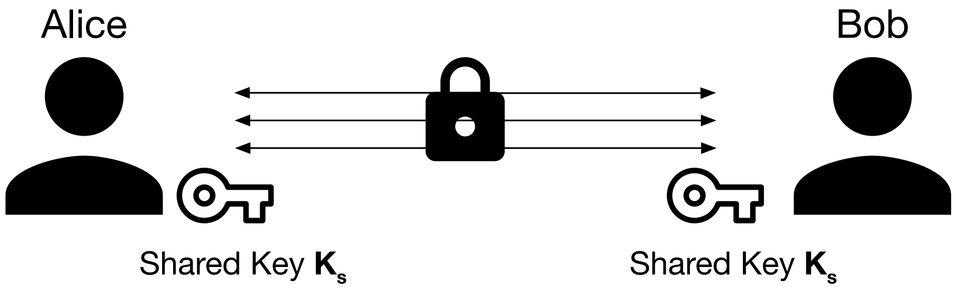
Overview: Problem of thousands of years



Problem

- Alice and Bob would like to communicate via unsecure channel
- Any 3rd person/party must not be able to understand the messages

Overview: Problem of thousands of years



- Confidentiality
  - Achieved: Only Alice and Bob who have the key can read the contents of the message
- Integrity
  - Achieved through seals: May be we can place a seal on the encrypted message?
- Authenticity
  - Achieved through seals: May be we can place a seal on the encrypted message?

Systems Security Lab

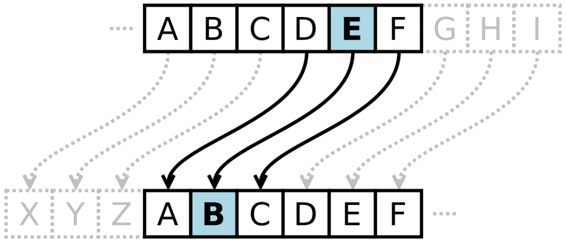
SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

History and Cryptography

- In 405 BC, Greek general Lysander of Sparta received coded message
- The message was successfully decoded by wrapping it around a wooden baton

History and Cryptography

- 2000 years ago
  - Caesar Cipher
    - Shift each letter forward by a fixed number *n*
    - Encode and decode by hand
- During World War I,II
  - Mechanical devices for encrypting and decrypting messages
- Today
  - Modern cryptography: rely on mathematics and electronic systems



Crypto Terminology

- Cryptology — The art and science of making and breaking "secret codes"
- Cryptography — making "secret codes"
- Cryptanalysis — breaking "secret codes"
- Crypto — all of the above (and more)

How to Speak Crypto

- A *cipher* or *cryptosystem* is used to *encrypt* the *plaintext*
- The result of encryption is *ciphertext*
- We *decrypt* ciphertext to recover plaintext
- A *key* is used to configure a cryptosystem
- A *symmetric key* cryptosystem uses the same key to encrypt as to decrypt
- A *public key* cryptosystem uses a *public key* to encrypt and a *private key* to decrypt

Crypto

- Basic assumptions
  - The system is completely known to the attacker
  - Only the key is secret
  - That is, crypto algorithms are not secret
- This is known as Kerckhoffs' Principle
- (Crypto version of *Open Design Principle* in Saltzer and Schroeder)
- Why do we make such an assumption?
  - Experience has shown that secret algorithms tend to be weak when exposed
  - Secret algorithms never remain secret
  - Better to find weaknesses beforehand

Systems Security Lab

Cryptoanalysis: An example on an ancient symmetric key cipher

Simple Substitution With Shifting

- A.K.A Caesar's cipher

Shift by k

a → c
b → d
c → e
…
z → b

Key k

$$c = E_k(\text{“abc”}),\ k = 2$$

$$= \text{“cde”}$$

Cryptanalysis of Caesar's Cipher

- We know that key is a fixed number *n*
- We know that the cipher algorithm E_k simply shifts each character by *n*
- How do we find the key?
  - Only 26 possible keys
  - Try them all – Exhaustive key search

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Substitution Cipher With Permutation

Use any permutation of letters

a → e
b → c
c → q
…
z → a

Key permutation

$$c = E_k(\text{“abc”})$$

$$= \text{“ecq”}$$

Then $26! > 2^{88}$ possible keys!

Cryptanalysis of Permutation Cipher

- Possible number of keys are $26! > 2^{88}$ Keys
- Exhaustive Key Search???

Cryptanalysis of Permutation Cipher

- Possible number of keys are $26! > 2^{88}$ Keys
- Exhaustive Key Search???
- NO

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Cryptanalysis of Permutation Cipher

- Given Ciphertext:

  PBFPVYFBQXZTYFPBFEQJHDXXQVAPTPQJKTOYQWIPBVWLXTOXBTFXQWAXBVCXQWAXFQJVWLEQNT
  OZQGGQLFXQWAKVWLXQWAEBIPBFXFQVXGTVJVWLBTPQWAEBFPBFHCVLXBQUFEVWLXGDPEQVPQ
  GVPPBFTIXPFHXZHVFAGFOTHFEFBQUFTDHZBQPOTHXTYFTODXQHFTDPTOGHFQPBQWAQJJTODXQ
  HFOQPWTBDHHIXQVAPBFZQHCFWPFHPBFIPBQWKFABVYYDZBOTHPBQPQJTQOTOGHFQAPBFEQJ
  HDXXQVAVXEBQPEFZBVFOJIWFFACFCCFHQWAUVWFLQHGFXVAFXQHFUFHILTTAVWAFFAWTEVOITD
  HFHFQAITIXPFHXAFQHEFZQWGFLVWPTOFFA

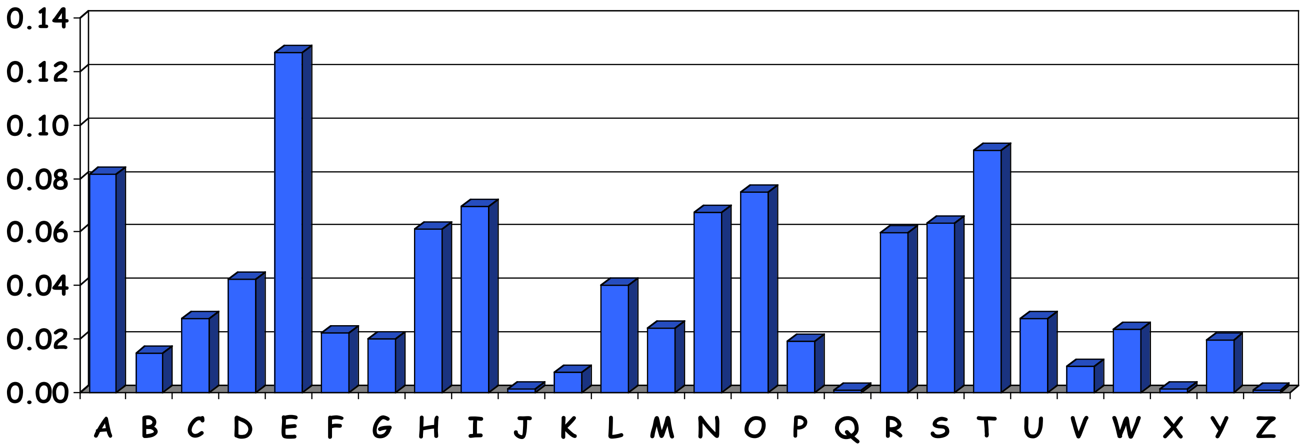
- Is there any statistical/mathematical property that can be our "shortcut" to the plaintext?

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Cryptanalysis II

- Cannot try all 2^{88} simple substitution keys
- Can we be more clever?
- English letter frequency counts…



Systems Security Lab

Cryptanalysis II

- Ciphertext:

PBFPVYFBQXZTYFPBFEQJHDXXQVAPTPQJKTOYQWIPBVWLXTOXBTFXQWAXBVCXQWAXFQJVWLEQNTOZQGGQLFX
QWAKVWLXQWAEBIPBFXFQVXGTVJVWLBTPQWAEBFPBFHCVLXBQUFEVWLXGDPEQVPQGVPPBFTIXPFHXZHVFAG
FOTHFEFBQUFTDHZBQPOTHXTYFTODXQHFTDPTOGHFQPBQWAQJJTODXQHFOQPWTBDHHIXQVAPBFZQHCFWP
FHPBFIPBQWKFABVYYDZBOTHPBQPQJTQOTOGHFQAPBFEQJHDXXQVAVXEBQPEFZBVFOJIWFFACFCCFHQWAUV
WFLQHGFXVAFXQHFUFHILTTAVWAFFAWTEVOITDHFHFQAITIXPFHXAFQHEFZQWGFLVWPTOFFA

Ciphertext frequency counts:

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
21	26	6	10	12	51	10	25	10	9	3	10	0	1	15	28	42	0	0	27	4	24	22	28	6	8

This is probably 'e' ???

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Cryptanalysis: Terminology

- Cryptosystem is secure if best know attack is to try all keys
  - Exhaustive key search, that is
- Cryptosystem is insecure if *any* shortcut attack is known
- But then insecure cipher might be harder to break than a secure cipher!
  - What the … ?

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Generalizing Symmetric Cipher Methodology

Systems
Security
Lab

SUNGKYUNKWAN UNIVERSITY
1398
성균관대학교

One-Time Pad: Encryption

e=000	h=001	i=010	k=011	l=100	r=101	s=110	t=111

Encryption: Plaintext ⊕ Key = Ciphertext

	h	e	i	l	h	i	t	l	e	r
Plaintext:	001	000	010	100	001	010	111	100	000	101
Key:	111	101	110	101	111	100	000	101	110	000
Ciphertext:	110	101	100	001	110	110	111	001	110	101
	s	r	l	h	s	s	t	h	s	r

Systems Security Lab

One-Time Pad: Decryption

e=000	h=001	i=010	k=011	l=100	r=101	s=110	t=111

Decryption: Ciphertext ⊕ Key = Plaintext

	s	r	l	h	s	s	t	h	s	r
Ciphertext:	110	101	100	001	110	110	111	001	110	101
Key:	111	101	110	101	111	100	000	101	110	000
Plaintext:	001	000	010	100	001	010	111	100	000	101
	h	e	i	l	h	i	t	l	e	r

Systems Security Lab

One-Time Pad

Double agent claims following "**key**" was used:

	s	r	l	h	s	s	t	h	s	r
Ciphertext:	110	101	100	001	110	110	111	001	110	101
"**key**":	101	111	000	101	111	100	000	101	110	000
"Plaintext":	011	010	100	100	001	010	111	100	000	101
	k	i	l	l	h	i	t	l	e	r

e=000 h=001 i=010 k=011 l=100 r=101 s=110 t=111

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

One-Time Pad

Or claims the key is...

	s	r	l	h	s	s	t	h	s	r
Ciphertext:	110	101	100	001	110	110	111	001	110	101
“key”:	111	101	000	011	101	110	001	011	101	101
“Plaintext”:	001	000	100	010	011	000	110	010	011	000
	h	e	l	i	k	e	s	i	k	e

e=000 h=001 i=010 k=011 l=100 r=101 s=110 t=111

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

One-Time Pad Summary

- Provably secure
  - Ciphertext gives no useful info about plaintext
  - All plaintexts are ***equally likely***
- BUT, only when be used correctly
  - Pad must be random, used only once
  - Pad is known only to sender and receiver
- Note: pad (key) is same size as message
- So, why not distribute msg instead of pad?

Codebook Cipher

- Literally, a book filled with “codewords”
- Zimmerman Telegram encrypted via codebook

Februar	13605
fest	13732
finanzielle	13850
folgender	13918
Frieden	17142
Friedenschluss	17149
:	:

- Modern block ciphers are codebooks!
- More about this later…

Codebook Cipher: Additive

- Codebooks also (usually) use additive
- Additive — book of "random" numbers
  - Encrypt message with codebook
  - Then choose position in additive book
  - Add in additives to get ciphertext
  - Send ciphertext and additive position (MI)
  - Recipient subtracts additives before decrypting
- Why use an additive sequence?

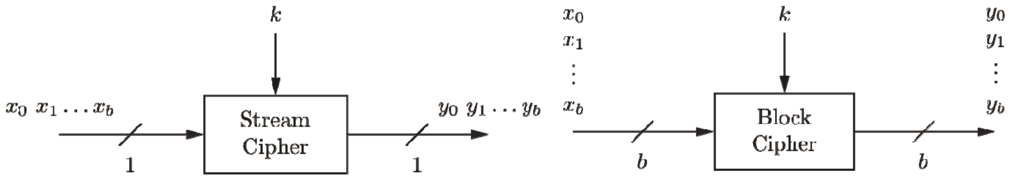
Systems Security Lab

Symmetric Key Crypto

- Stream cipher — generalize one-time pad
  - Except that key is relatively short
  - Key is stretched into a long keystream
  - Keystream is used just like a one-time pad
- Block cipher — generalized codebook
  - Block cipher key determines a codebook
  - Each key yields a different codebook
  - Employs both “confusion” and “diffusion”

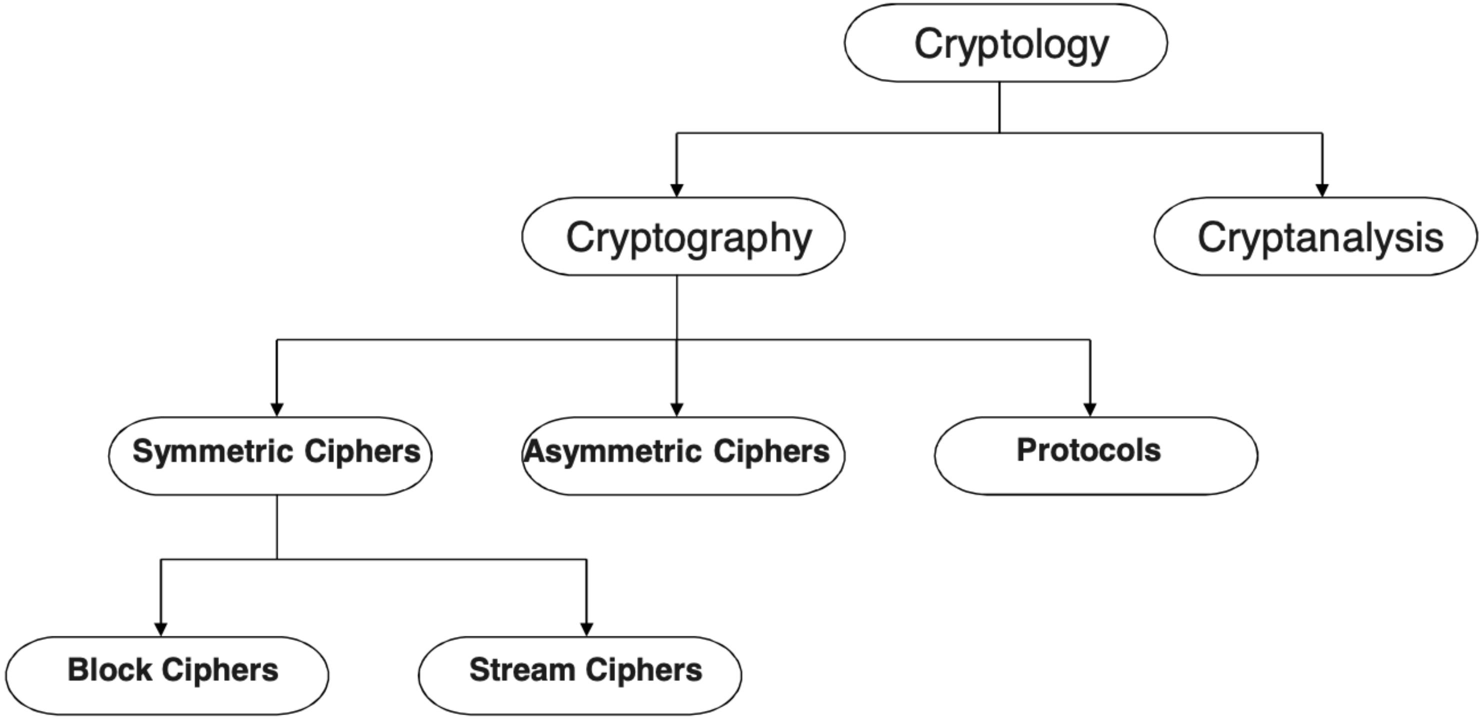
Systems Security Lab

Stream Ciphers vs. Block Ciphers



- Stream Ciphers
  - Encrypt bits individually
  - Small and fast, easier to implement in hardware
- Block Ciphers
  - Always encrypt in blocks (N bits)
  - Better for high data throughput
    - Cost of ciphers amortized during data processing

Modern Cryptography



Modern Cryptography

- Symmetric-Key Cryptography (a.k.a. Shared Key …)
  - The same secret key is used by both sides of a communication
  - Stream Cipher
  - Block Cipher
- Asymmetric-Key Cryptography (a.k.a Public-Key …)
  - The two sides of a communication uses different keys
- Hash Algorithms
  - Special symmetric and unkeyed crypto
    - One way encryption

One more thing: Randomness

Does Randomness Truly Exist In Our Universe?

A hint for your next CTF challenge

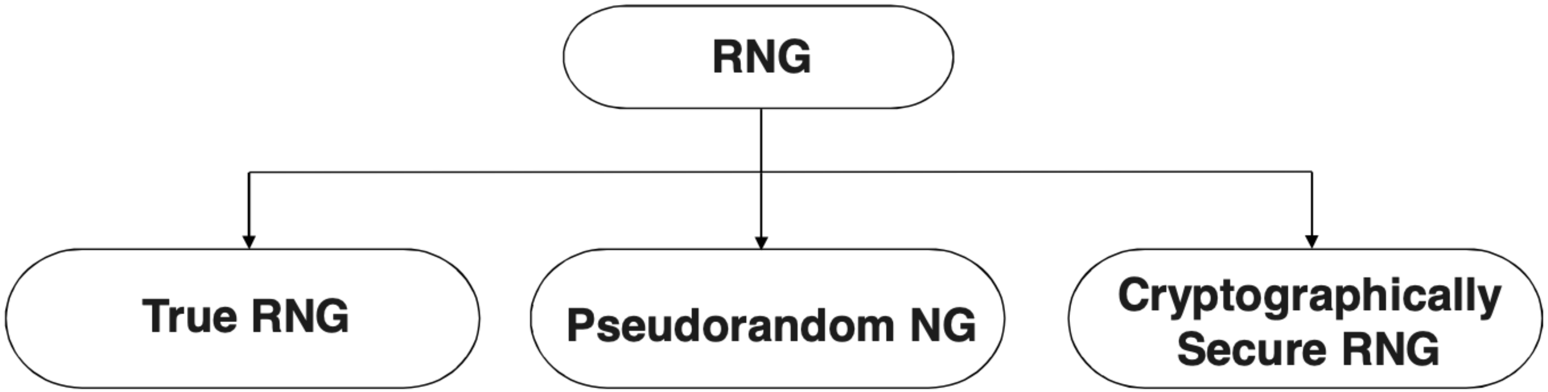
One more thing: Randomness

- I gave you the hint, so please don't bruteforce on the next challenge
- 50+ ppl bruteforcing may become a DDoS attack.

One more thing: Randomness

- Based on what we learned, and also based on how modern crypto works,
- Having a *reliable* source of randomness is critical for configuring and using cryptosystems.
- How do we obtain random numbers?

One more thing: Randomness



True Random Number Generators (TRNGs)

- Based on physical randomness
  - Semiconductor noise, jitters in digital circuits, etc..
  - Statiscally random
- Often implemented in hardware to provide random number generators to software

Pseudorandom Number Generators (PRNGs)

- Generate sequences from initial *seed* value
- Given seed, it can create *stream* of statistically random bytes
- EX)

Example: *rand() function* in ANSI C:

$$s_0 = 12345$$

$$s_{i+1} = 1103515245 s_i + 12345 \bmod 2^{31}$$

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Pseudorandom Number Generators (PRNGs)

- If attacker can somehow influence or learn of your seed, your cryptosystem becomes *predictable*

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Cryptographically Secure Pseudorandom Number Generators (CSPRNGs)

- Special PRNG with secure properties
  - Output must be *unpredictable*
- Given n consecutive bits of output S_i consecutive bits , S_{n+1} must not be predictable

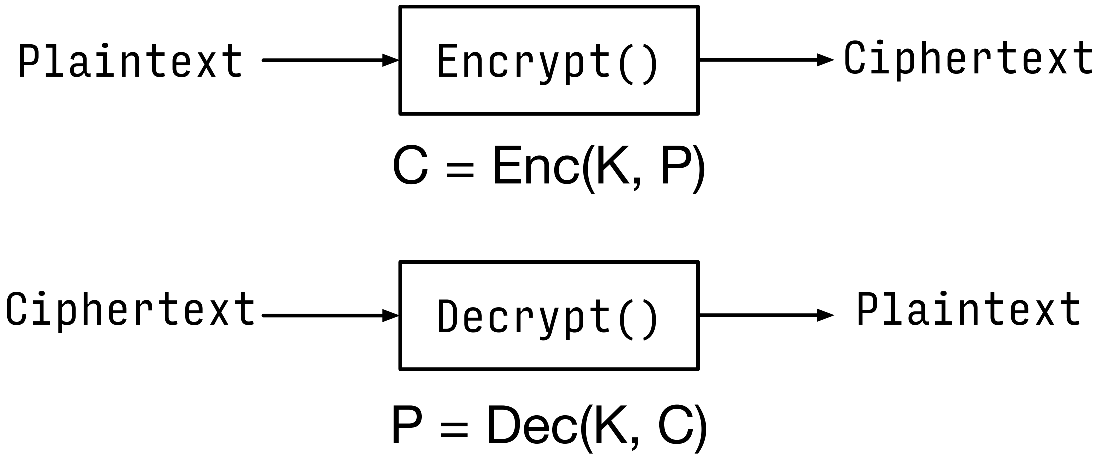
Systems Security Lab

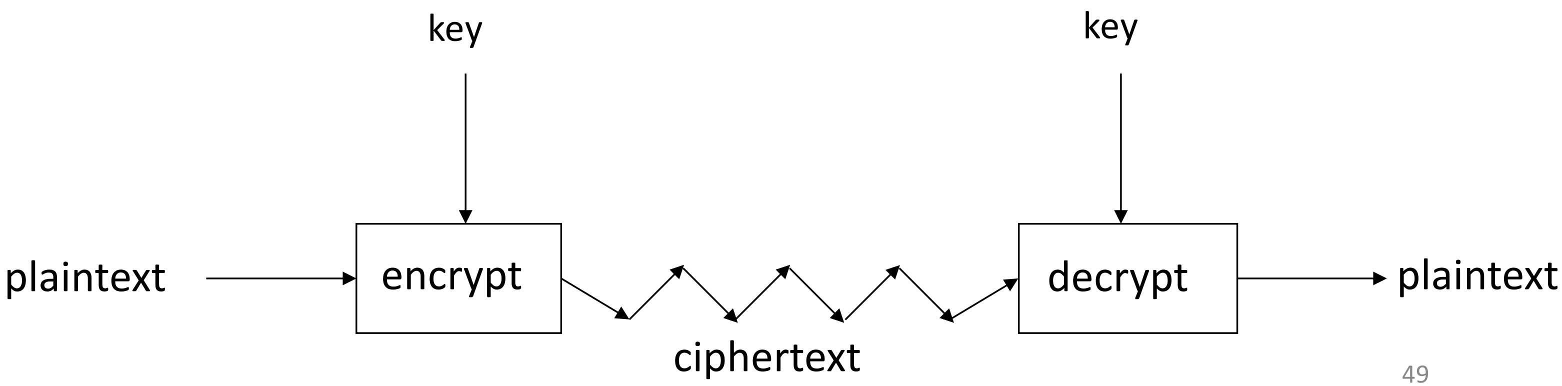
SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

Modern Symmetric-Key Cryptography

Symmetric-Key Cryptography

- Uses the same key for encryption/decryption
- Assumption: Sender and Receiver already have a shared secret key





Symmetric-Key Cryptography

- Data Encryption Standard (DES)
  - Developed by the IBM with influence of Nation Security Agency (NSA)
  - Block Cipher
    - Also can be configured to use Stream Cipher
    - Block size: 64 bits
  - Key size: 56 bit

Symmetric-Key Cryptography

- Data Encryption Standard (DES)
  - Standardized in 1977 by the NIST (National Institute of Standards and Technology)
  - Most popular block cipher for 30+ years (1970s~2000s)
  - Very well-studied algorithm
  - Due to the *short key length* (56bit), considered *insecure* nowadays

Symmetric-Key Cryptography

- Data Encryption Standard (DES)
  - 3DES (uses three DES keys) came out
  - However, AES is more efficient and safer

Symmetric-Key Cryptography

- Advanced Encryption Standard (AES)
  - Based on original block cipher developed by two Belgian cryptographers Vincent Rijmen and Joan Daemen
  - Adopted by NIST
  - Block Cipher:
    - Blocksize: 128bits
  - Key sizes: 128,192, 256, 512 bits

Symmetric-Key Cryptography

- Advanced Encryption Standard (AES)
  - Based on original block cipher developed by two Belgian cryptographers Vincent Rijmen and Joan Daemen
  - Adopted by NIST
  - Stream Cipher:
    - Blocksize: 128bits
  - Key sizes: 128,192, or 256 bits

Symmetric-Key Cryptography

- Advanced Encryption Standard (AES)
  - Considered a gold standard these days,
  - Implemented in hardware in most modern architectures
    - Intel's AES-NI, ARM etc..

Symmetric-Key Cryptography

Block Cipher Modes of Operation

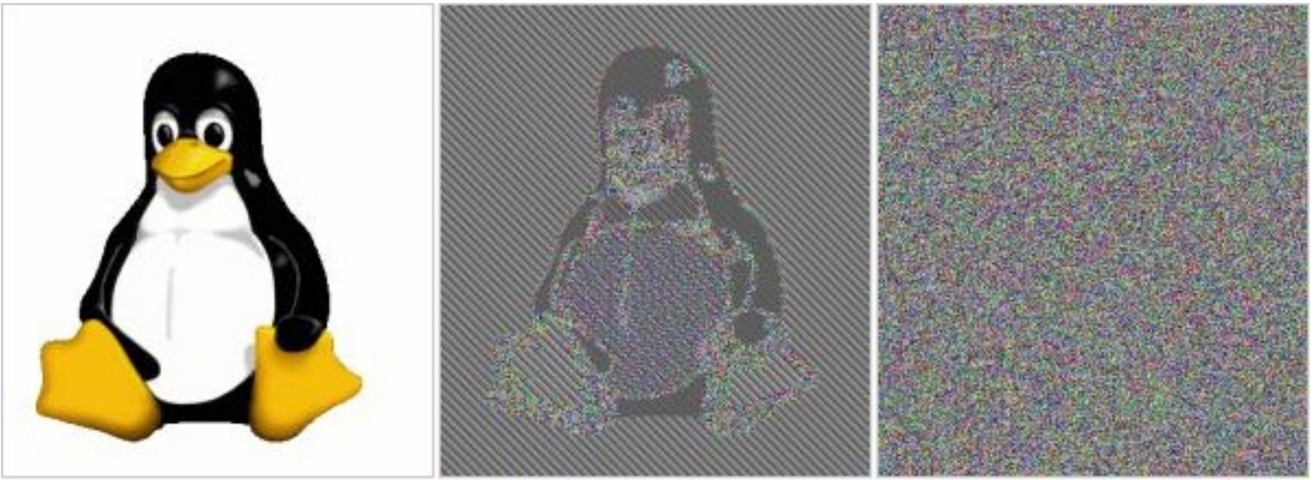
- Electronic Code Book (ECB) mode
- Cipher Block Chaining (CBC) mode
- Output Feedback mode (OFB) mode
- Cipher Feedback mode (CFB) mode
- Counter mode (CTR)
- Galois Counter Mode (GCM)

Symmetric-Key Cryptography

- There are many ways of encrypting long plaintexts (e.g., files and network streams)
- There can be multiple *modes* of operation that
  - Provide authenticity and integrity to confidentiality

Systems Security Lab

Symmetric-Key Cryptography



Original image | Encrypted using ECB mode | Modes other than ECB result in pseudo-randomness

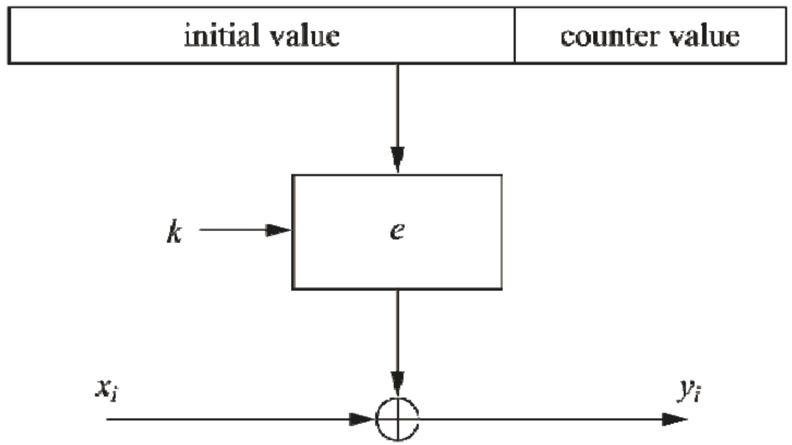
Case study: ECB

- Ciphertexts are recognizable!

Symmetric-Key Cryptography

Case study: CTR

- For every block, counter values are added to compute a new key stream block



Symmetric-Key Cryptography

Case study: GCM

- C in GCM is “Counter”
  - Same ciphertext diversification scheme as CTR
- GCM also computes MAC (message authentication code) during encryption
  - Message authentication
    - Message was created by the original sender (who has symmetric key)
  - Message integrity
    - Message was not modified during transmission

Symmetric-Key Cryptography

Case study: GCM

- GCM also computes MAC (message authentication code) during encryption
  - Message authentication
    - Message was created by the original sender (who has symmetric key)
  - Message integrity
    - Message was not modified during transmission

Systems Security Lab

SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

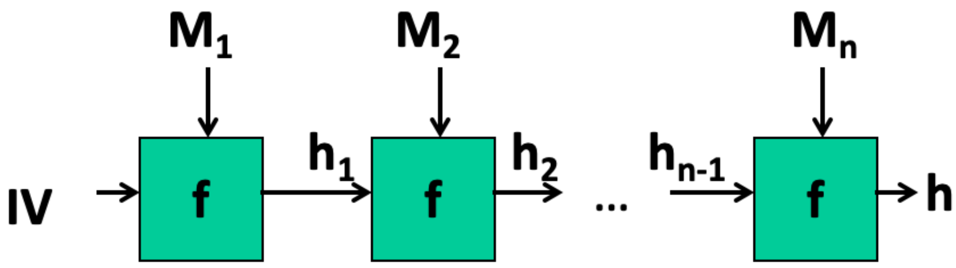
Symmetric-Key Cryptography

- A new challenger: Chacha20
  - Stream cipher developed by Google
  - Considered as safe as AES and included in TLS 1.2, 1.3 (We'll discuss this later)
  - AES is generally faster on high-end machines whereas Chacha is faster on mobile and embedded devices.

Systems Security Lab

Hash Functions

- Hashing is a one-way only encryption
  - No such thing as unhashing or dehashing
- There is <u>no key</u> used in hashing
  - $H(m) == h$ vs. $Enc(key_{enc}, m) = c$
- Fast computation time



Hash Functions

- Purpose: produce a fixed-size "fingerprint" or digest of arbitrarily long input data
- Hash passwords such that password plaintext need not be saved on the service or server
- <u>To guarantee integrity</u>

Hash Functions

Thank you for downloading Ubuntu Desktop

Your download should start automatically. If it doesn't, download now.

You can verify your download, or get help on installing.

Run this command in your terminal in the directory the iso was downloaded to verify the SHA256 checksum:

```
echo
"c0d025e560d54434a925b3707f8686a7f588c42a5fbc609b8ea2447f8884
7041 *ubuntu-18.04.4-desktop-amd64.iso" | shasum -a 256 --
check
```

You should get the following output:

```
ubuntu-18.04.4-desktop-amd64.iso: OK
```

Or follow this tutorial to learn how to verify downloads

Help

and u

Com

3

Systems Security Lab

MAC

- Message Authentication Code (MAC)
- One-way Function (Basically a Hash function with a key) that creates a message *digest*
  - e,g, MAC(k,m) = d
- A digest is appended at the end of the message, so that the receiver can verify it

Systems Security Lab

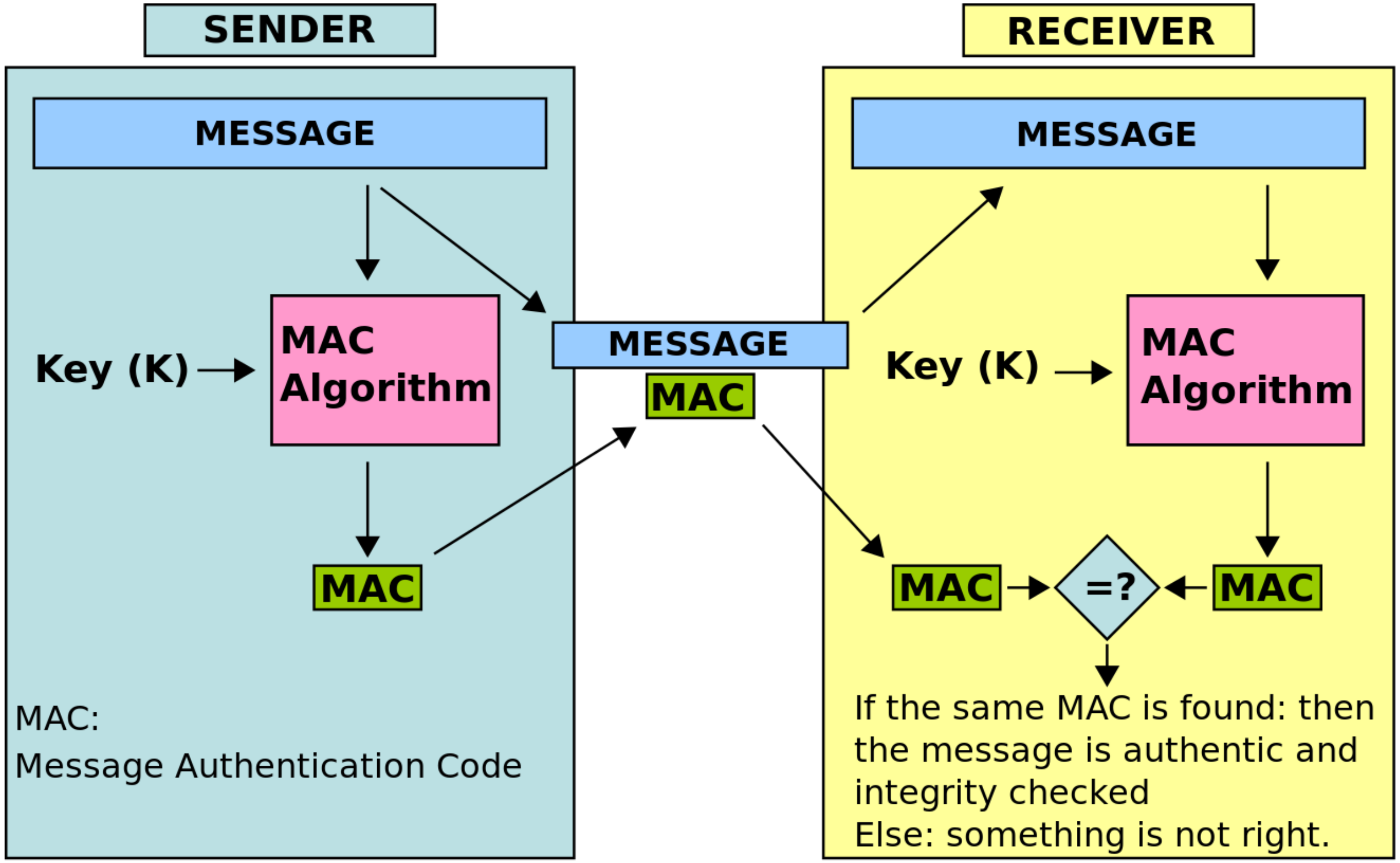
SUNGKYUNKWAN UNIVERSITY 1398 성균관대학교

MAC vs Hash

- Key is used during computation
- Ensures integrity and authenticity of the message
- A shared key is need to verify a MAC

- Key is not used during computation
- Ensures only integrity
- Everyone can verify a hash

MAC



Systems Security Lab

HMAC

- Hash-based Message Authentication Code (HMAC)
- Most widely used form of MAC today
- Builds a MAC out of hash functions (e.g., SHA-256)

$$\mathrm{HMAC}(K, m) = \mathrm{H}\Big(\big(K' \oplus opad\big) \parallel \mathrm{H}\big(\big(K' \oplus ipad\big) \parallel m\big)\Big)$$

$$K' = \begin{cases} \mathrm{H}(K) & K \text{ is larger than block size} \\ K & \text{otherwise} \end{cases}$$

H is a cryptographic hash function
m is the message to be authenticated
K is the secret key
K' is a block-sized key derived from the secret key

Systems Security Lab

Summary

- MACs are One way functions that takes a key and a message and creates a message digest
  - Integrity
  - Authenticity
- The digest is usually appended at the end of the message so that the receiver can verify it
- HMAC turns hash functions into MACs and widely used today