CS 370: Introduction to Security
04.11: Block-cipher and symmetric enc.

Tu/Th 4:00 – 5:50 PM (Recording)

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

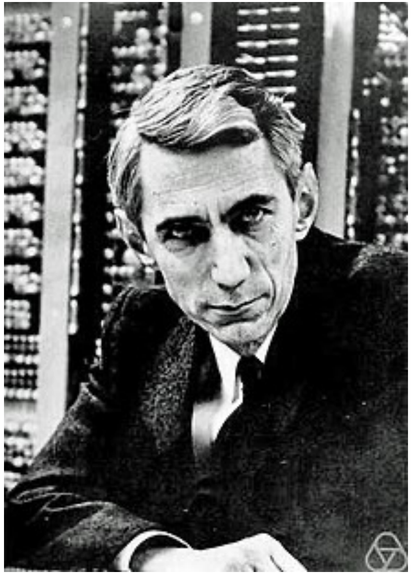
SAIL
Secure AI Systems Lab

Topics for today

- Recap
  - Perfect security (XOR)
  - XOR's practical limitations
  - Stream ciphers (RC4/5)
- Block ciphers
  - What is the block cipher?
  - How does the block cipher work?
  - How secure are the block ciphers?
- Symmetric encryptions
  - What are DES and AES?
  - What is ECB and how does it work?
  - What are the weaknesses in ECB?
  - How can an adversary exploit it (Micro-labs)?

Recep: perfect security

- Shannon's intuition
  - An adversary should not distinguish a message M from a random text R
  - Formally:
    - Pr[M = m|C = c] = Pr[M = m]
    - where
      - m is a message (from a set M)
      - c is a ciphertext (from a set of all ciphertexts C)
    - Pr[C = c|M = m] = Pr[C = c]
  - It means:
    - Ciphertext provides no additional information
    - Observing c does not help with guessing M = m
    - c is independent of the message m



Claude Shannon (1916 ~ 2001)
A Father of Information Theory
and Modern Cryptography

Recap: XOR cipher

- Crypto scheme with perfect secrecy
  - Encryption:
    - Given a message *m* and a random key *k*
    - Ciphertext $c = m \oplus k$
  - Example:
    - Message: HELLO
    - Key : ABCDE

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Message	H (0x48)	E (0x45)	L (0x4c)	L (0x4c)	O (0x4f)
Key	A (0x41)	B (0x42)	C (0x43)	D (0x44)	E (0x45)
Ciphertext	**0x9**	**0x7**	**0xf**	**0x8**	**0xa**

Recap: XOR cipher

- Crypto scheme with perfect secrecy
  - Encryption:
    - Given a message *m* and a random key *k*
    - Plaintext $m = k \oplus c$
  - Example:
    - Message: HELLO
    - Key : ABCDE

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

Key	A (0x41)	B (0x42)	C (0x43)	D (0x44)	E (0x45)
Ciphertext	0x9	0x7	0xf	0x8	0xa
Decrypt	**H**	**E**	**L**	**L**	**O**

Recap: XOR cipher: in bitwise operation

- Example from Wikipedia[1]

The string "Wiki" (01010111 01101001 01101011 01101001 in 8-bit ASCII) can be encrypted with the repeating key 11110011 as follows:

```
  01010111 01101001 01101011 01101001
⊕ 11110011 11110011 11110011 11110011
  -----------------------------------
= 10100100 10011010 10011000 10011010
```

And conversely, for decryption:

```
  10100100 10011010 10011000 10011010
⊕ 11110011 11110011 11110011 11110011
  -----------------------------------
= 01010111 01101001 01101011 01101001
```

A	B	Q
0	0	0
0	1	1
1	0	1
1	1	0

[1]Image from: https://en.wikipedia.org/wiki/XOR_cipher

Recap: XOR cipher

- Potential problems:
  - If an adversary knows a pair of *m* and *c*
    - They can extract the key by $m \oplus c$
    - We should not use the same key multiple times (use OTP)
  - (Practical issue) The length of the *k* should be the same as *m*
    - What if we want to encrypt a 1GB video file?
    - We need to use a 1GB key (even multiple of those)
    - How can we share the 1GB keys with others?

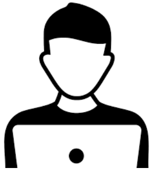
Recap: stream cipher

- Reduce key generation and exchange overheads
  - Encryption:
    - Given a message *m* and a random key *k*
    - Ciphertext $c = m \oplus k$
    - and:
      - The key stream is generated by the same mechanism for a sender and a receiver
      - The key stream is a byte stream (0xAB129dB…)
      - It performs XOR encryption over this byte stream

Recap: Stream cipher

- Stream cipher
  - Example:

Encrypt message 1 with 0x1b395a46
Encrypt message 2 with 0xf1737202
Encrypt message 3 with 0xccf0de05…



A random number generator

1: 0x1b395a46
2: 0xf1737202
3: 0xccf0de05
4: 0x908b0feb
5: 0x9d4c9add
…

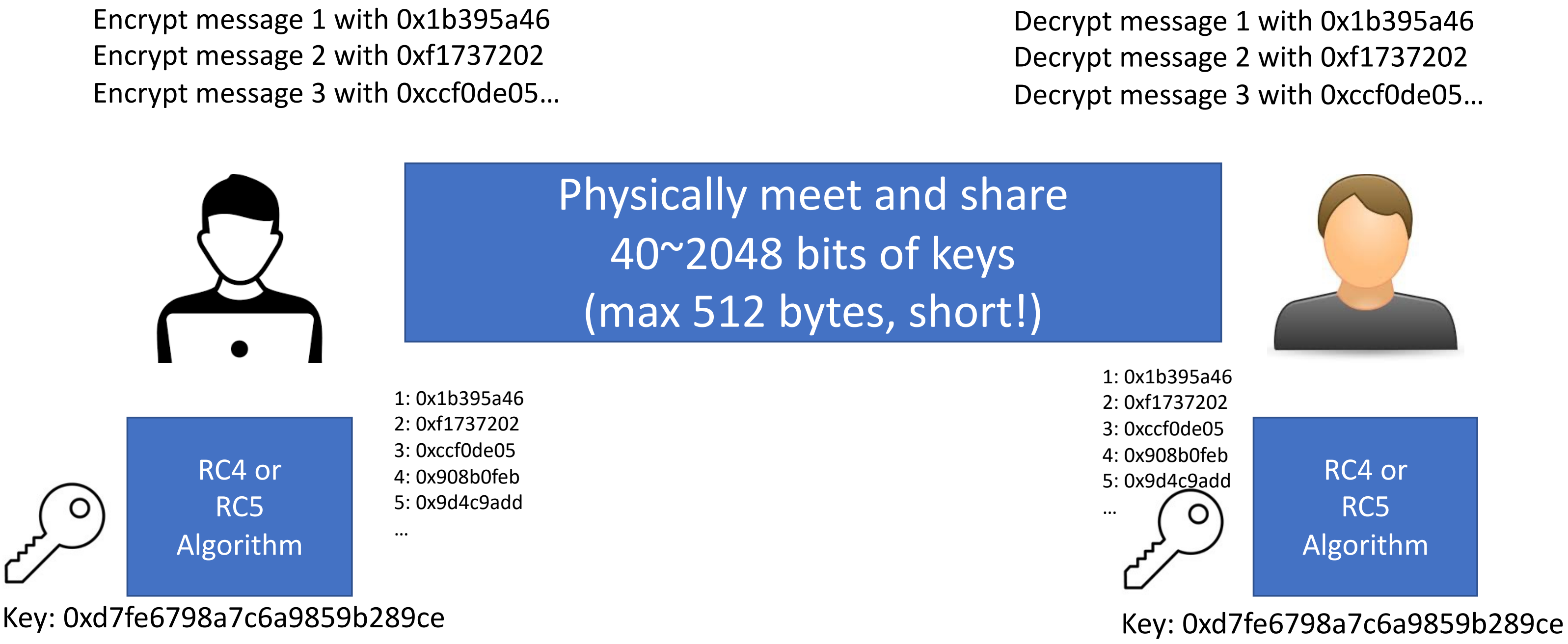
Decrypt message 1 with 0x1b395a46
Decrypt message 2 with 0xf1737202
Decrypt message 3 with 0xccf0de05…

1: 0x1b395a46
2: 0xf1737202
3: 0xccf0de05
4: 0x908b0feb
5: 0x9d4c9add
…

A random number generator

Recap: RC4/5 stream cipher

- Stream cipher
  - Example: RC4/RC5



Recap: RC4/5 stream cipher

- Potential problems
  - Have no mathematical proof[1]
  - Have seen their vulnerabilities to
    - Bit-flipping attacks
    - Reused key attacks
    - Differential attacks
    - …

[1]https://en.wikipedia.org/wiki/RC4#Security

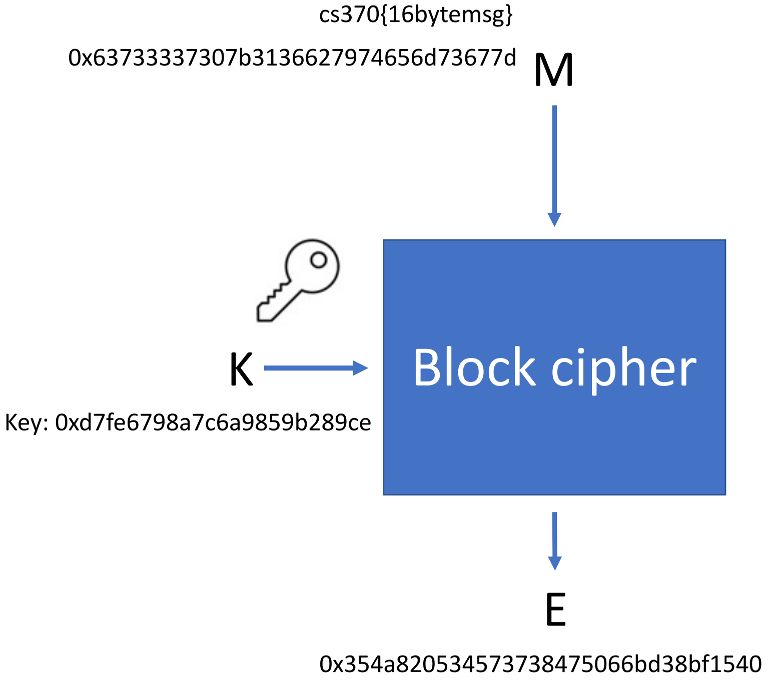
Topics for today

- Recap
  - Perfect security (XOR)
  - XOR's practical limitations
  - Stream ciphers (RC4/5)
- Block ciphers
  - What is the block cipher?
  - How does the block cipher work?
  - How secure are the block ciphers?
- Symmetric encryptions
  - What are DES and AES?
  - What is ECB and how does it work?
  - What are the weaknesses in ECB?
  - How can an adversary exploit it (Micro-labs)?

Block cipher: encryption

- Block cipher
  - Cryptographic algorithm that work only with fixed-length set of bits

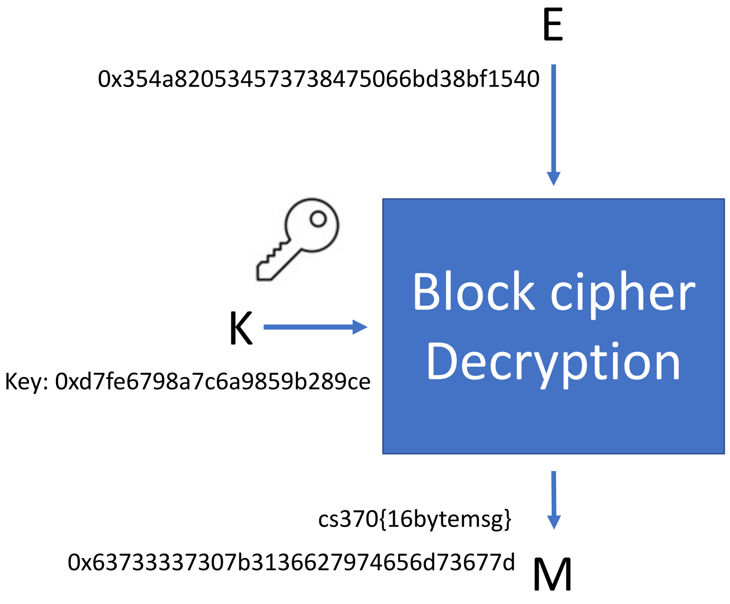
- Terminology
  - **Block:** a fixed size message *M*
  - **Key:** a secret we use for encryption
    - Shared between a sender and a receiver
  - **Encryption:** use *K* to convert *M* into *E*



Block cipher: decryption

- Block cipher
  - Cryptographic algorithm that work only with fixed-length set of bits

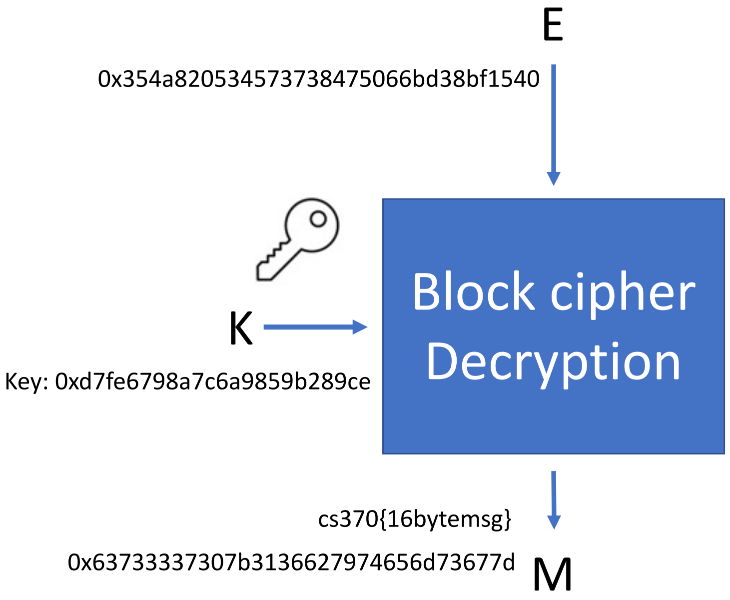
- Terminology
  - **Block:** a fixed size message *M*
  - **Key:** a secret we use for encryption
    - Shared between a sender and a receiver
  - **Encryption:** use *K* to convert *M* into *E*
  - **Decryption:** use *K* to convert *E* into *M*



Block cipher

- Block cipher
  - Cryptographic algorithm that work only with fixed-length set of bits

- Terminology
  - **Block:** a fixed size message *M*
  - **Key:** a secret we use for encryption
    - Shared between a sender and a receiver
  - **Encryption:** use *K* to convert *M* into *E*
  - **Decryption:** use *K* to convert *E* into *M*

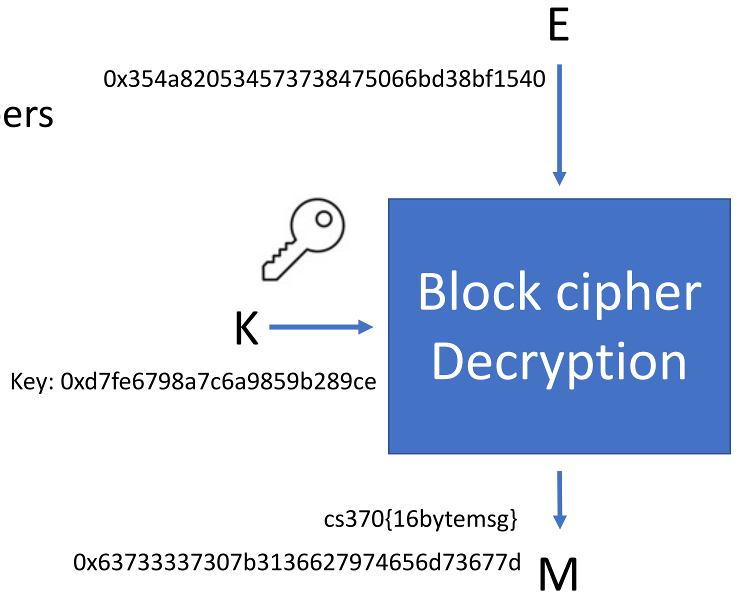


- Formally
  - You can see encryption and decryption as
  - Generating a permutation of numbers:
  - $\{0,1\}^n \rightarrow \{0,1\}^n$ (1-to-1 mappings)

Block cipher

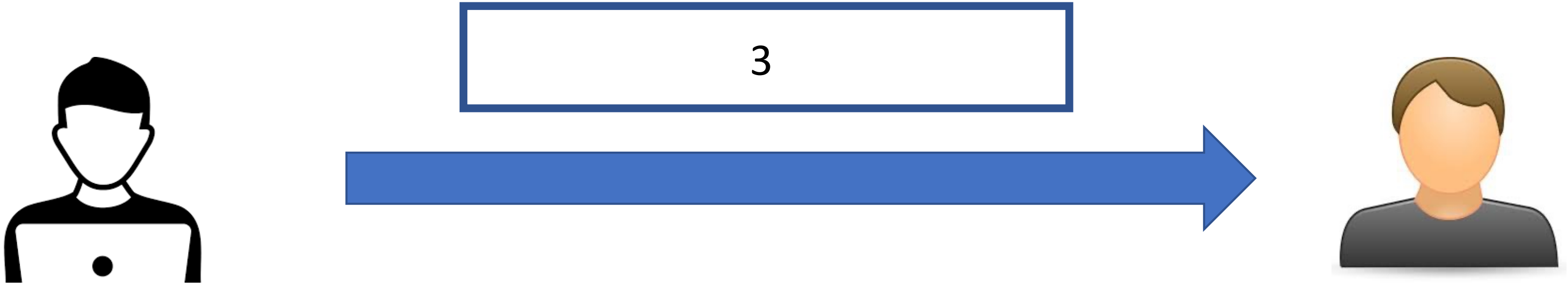
- Formally
  - You can see encryption and decryption as
  - Generating a permutation of numbers:
    - $\{0,1\}^n \rightarrow \{0,1\}^n$
    - Mappings should be 1-to-1
  - The key determines how to permute the numbers

M	Ciphertext
0	0xaf531b0e1
1	0x14a986e7a
2	0xad738009d
3	0x5ed6985c5
4	0xf3b8aa2e8
5	0xad04ec00e
…	0x59fd94c21



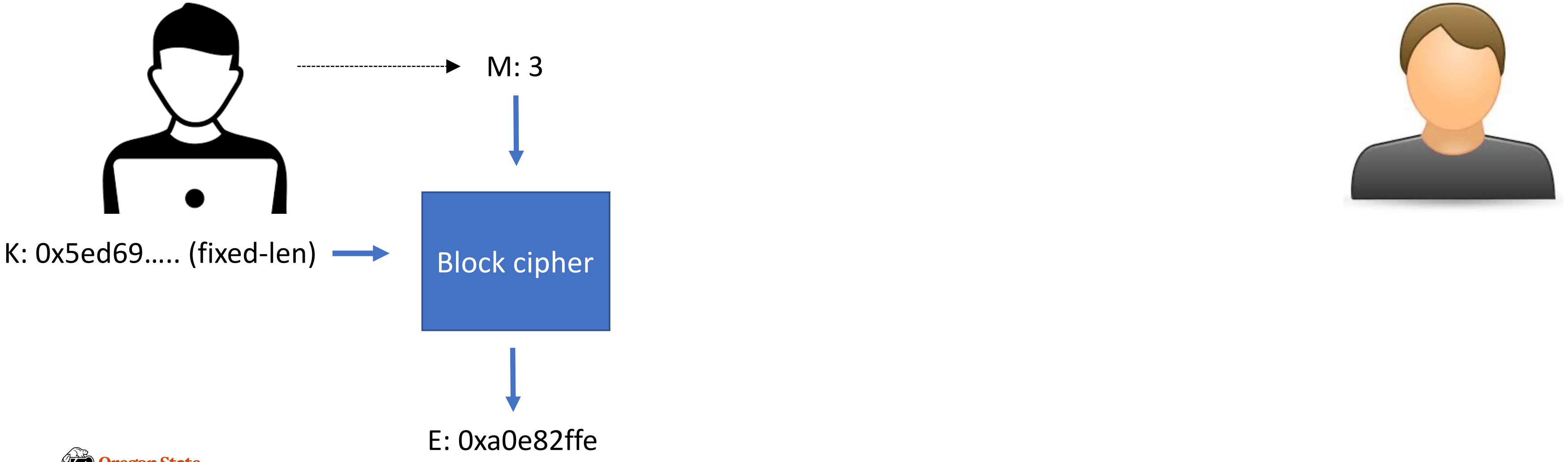
Block cipher: in operation

- Goal
  - We want to communicate with others securely (and privately)



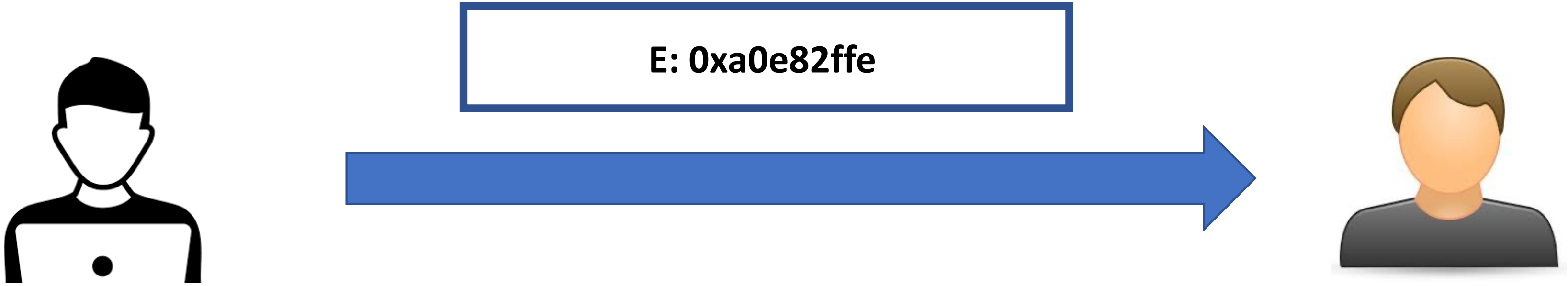
Block cipher: in operation

- Goal
  - We want to communicate with others securely (and privately)
  - Both parties use the same block cipher algorithm
    - 1st: Share the information about the key to use



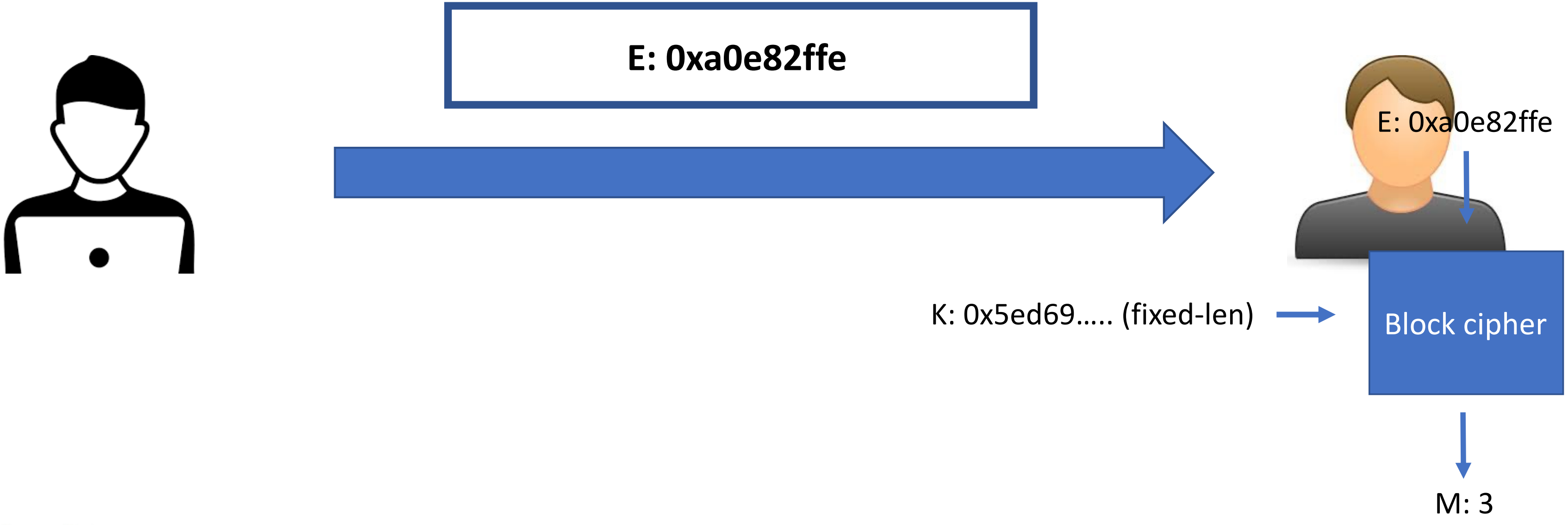
Block cipher: in operation

- Goal
  - We want to communicate with others securely (and privately)
  - Both parties use the same block cipher algorithm
    - 1st: Share the information about the key to use



Block cipher: in operation

- Goal
  - We want to communicate with others securely (and privately)
  - Both parties use the same block cipher algorithm
    - 1st: Share the information about the key to use



Block cipher: in operation

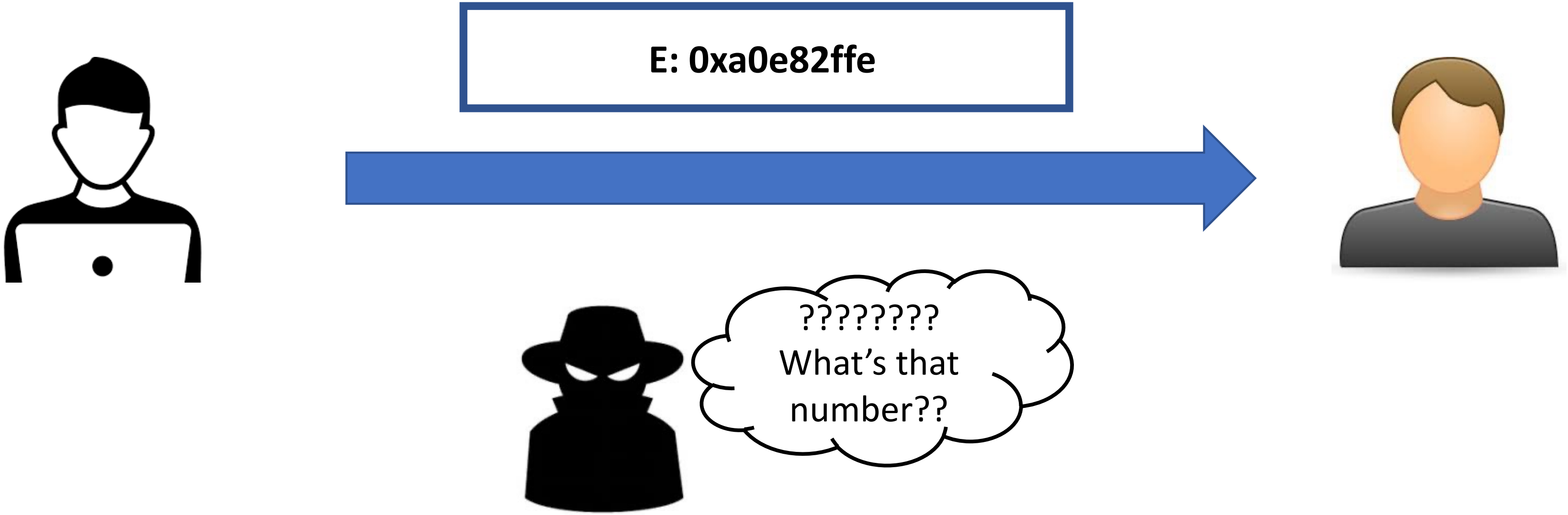
- Goal
  - We want to communicate with others securely (and privately)
  - Both parties use the same block cipher algorithm
    - 1st: Share the information about the key to use

M	Ciphertext
0	0x87372de1
1	0x19f1a578
2	0x9449fe68
3	0xa0e82ffe
4	0xba9f4d4b
5	0x5156b9ba
…	0x61e0bb0d

M	Ciphertext
0	0x87372de1
1	0x19f1a578
2	0x9449fe68
3	0xa0e82ffe
4	0xba9f4d4b
5	0x5156b9ba
…	0x61e0bb0d

Block cipher: in operation

- Goal
  - We want to communicate with others securely (and privately)
  - Both parties use the same block cipher algorithm
    - 1st: Share the information about the key to use



Topics for today

- Recap
  - Perfect security (XOR)
  - XOR's practical limitations
  - Stream ciphers (RC4/5)
- Block ciphers
  - What is the block cipher?
  - How does the block cipher work?
  - How secure are the block ciphers?
- Symmetric encryptions
  - What are DES and AES?
  - What is ECB and how does it work?
  - What are the weaknesses in ECB?
  - How can an adversary exploit it (Micro-labs)?

Block cipher: security

- (Ideal) Block ciphers
  - Must be a random permutation of messages
  - True randomness is difficult to achieve
    - We need a scheme like one-time pad (private random source)
    - It’s computationally demanding (we need to carry all the time, we should choose…)

Block cipher: security

- (Ideal) Block ciphers
  - Must be a random permutation of messages
  - True randomness is difficult to achieve
    - Security: we assume the adversary knows everything
    - Resources: we need one-time pads for everything
- Pseudo-randomness
  - It is not truly random
  - But it is indistinguishable from the true randomness
  - Generator that deterministically outputs that look like random
  - An attacker needs a lot of effort to guess the number (e.g., in AES, it requires $2^{126.1}$ guesses)

1: 0x1b395a46
2: 0xf1737202
3: 0xccf0de05
4: 0x908b0feb
5: 0x9d4c9add
…

A pseudo- random number generator

Block cipher: security

- $2^{126.1}$ computations
  - 9.1176402658 x 10^{37} times
  - Suppose that we need 1 CPU cycle to guess once
    - 6GHz CPU, the time it takes: 9.1176402658 x 10^{37} / 6 x 10^{9}
    - = 1.5196 x 10^{28} seconds
    - = 481,860,000,000,000,000,000 years
    - = 481,860,000,000 years with 1 billion super-computer

1: 0x1b395a46
2: 0xf1737202
3: 0xccf0de05
4: 0x908b0feb
5: 0x9d4c9add
…

A pseudo- random number generator

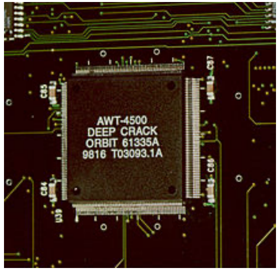
Topics for today

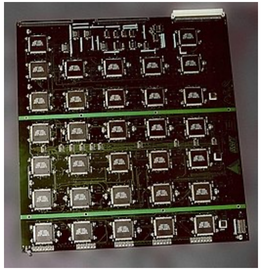
- Recap
  - Perfect security (XOR)
  - XOR's practical limitations
  - Stream ciphers (RC4/5)
- Block ciphers
  - What is the block cipher?
  - How does the block cipher work?
  - How secure are the block ciphers?
- Symmetric encryptions
  - What are DES and AES?
  - What is ECB and how does it work?
  - What are the weaknesses in ECB?
  - How can an adversary exploit it (Micro-labs)?

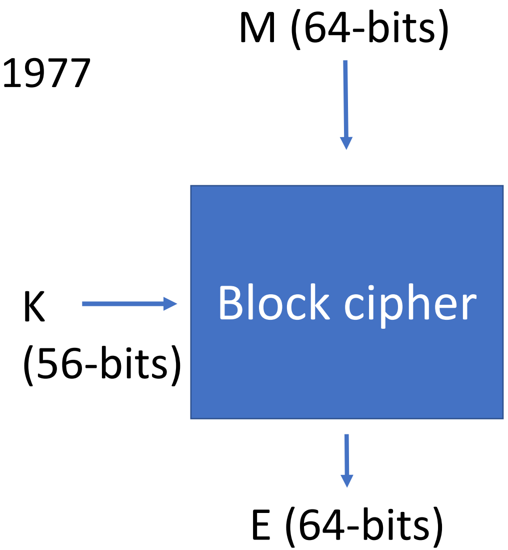
Symmetric encryption

- Symmetric encryption
  - Encryption that uses the same key for encrypting and decrypting

- DES
  - Data Encryption Standard (developed by IBM in early 1970)
  - Becomes Federal Information Processing Standard (FIPS) in 1977
  - Uses 56-bit key
    - At that time brute-forcing 56-bit key was difficult
    - But broken by Electronic Frontier Foundation (EFF) in 1999



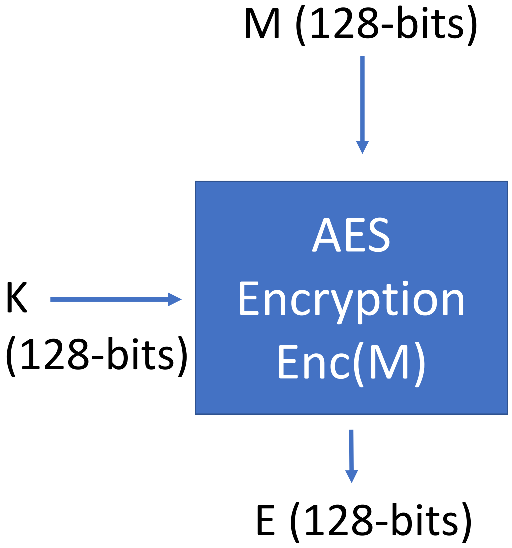




Symmetric encryption – cont'd

- Symmetric encryption
  - Encryption that uses the same key for encrypting and decrypting

- AES
  - Advanced Encryption Standard
  - Key size: 128-/192-/256-bit
  - Block size: 128-bit (16-byte)
  - Ciphertext and message size: 128-bit



Electronic code block

- ECB
  - A mode of block cipher operations
  - We pad the length of a message at the end

- ECB Operation
  - Suppose that we encrypt 15-byte data: 0123456789ABCDE (e.g., 0 = 0x30)
  - ECB pads 0x01 (= 1-byte length) at the end

Pos	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
Hex	0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x41	0x42	0x43	0x44	0x45	0x01

Electronic code block – cont'd

- ECB
  - A mode of block cipher operations
  - We pad the length of a message at the end

- ECB Operation
  - Suppose that we encrypt 14-byte data: 0123456789ABCD (e.g., 0 = 0x30)
  - ECB pads 0x02 x 2 (= 2-byte length) at the end

Pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x41	0x42	0x43	0x44	0x02	0x01

Electronic code block – cont'd

- ECB
  - A mode of block cipher operations
  - We pad the length of a message at the end

- ECB Operation
  - Suppose that we encrypt 1-byte data: 0 (e.g., 0 = 0x30)
  - ECB pads 0x0F x 15 (= 15-byte length) at the end

Pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0x30	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x0F	0x02	0x01

Electronic code block – cont'd

- ECB
  - A mode of block cipher operations
  - We pad the length of a message at the end

- ECB Operation (corner-case)
  - Suppose that we encrypt 16-byte data: 0123456789ABCDE\x01 (e.g., 0 = 0x30)
  - How we can distinguish this from 15-byte data with 0x01 padding
  - We pad 16-byte of 0x10 at the end (= we encrypt two blocks)

Pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0x30	0x31	0x32	0x33	0x34	0x35	0x36	0x37	0x38	0x39	0x41	0x42	0x43	0x44	0x45	0x01

Pos	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Hex	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10	0x10

Electronic code block – cont'd

- ECB
  - A mode of block cipher operations
  - We pad the length of a message at the end

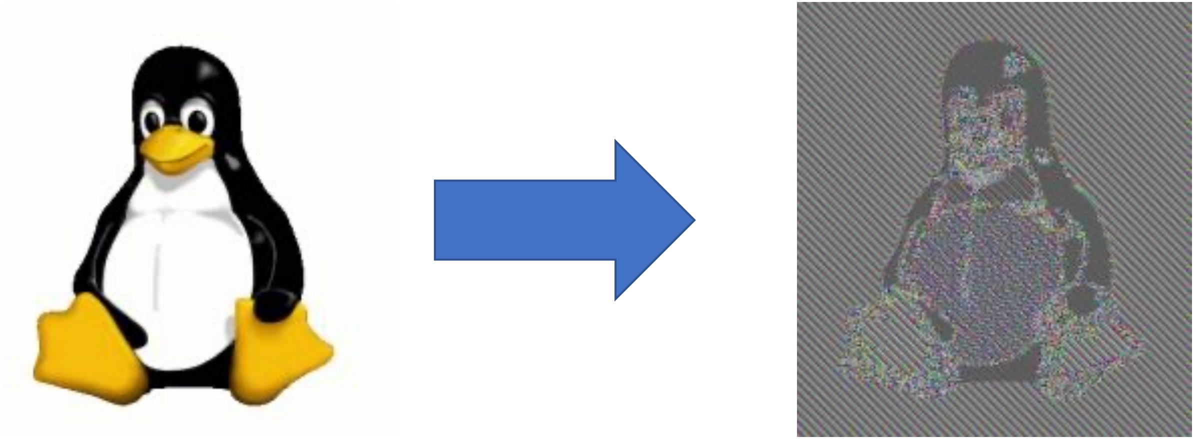
- ECB Operation
  - Suppose that we encrypt 31-byte data: 0123456789ABCDEF0123456789ABCDE
  - How can we encrypt/decrypt this message?
    - Split the message into 16-bytes: 0123456789ABCDEF + 0123456789ABCDE
    - Encrypt the first block: 0123456789ABCDEF
    - Encrypt the second block (with pads): 0123456789ABCDE\x01

Electronic code block – cont'd

- ECB weakness(es)
  - Using the same key leads to the same ciphertext
  - An adversary can guess the message by looking at the ciphertext
  - Suppose:
    - M: 0 -> C: 0x39827332…
    - M: 1 -> C: 0x5a83f874…
    - …

Micro-labs

- ECB weakness
  - I will provide you a super-secretly-encrypted photo
  - Your job is to guess what’s in the photo
  - This is my encryption algorithm



Thank You!

Tu/Th 4:00 – 5:50 PM (Recording)

Sanghyun Hong

sanghyun.hong@oregonstate.edu

Oregon State University

SAIL
Secure AI Systems Lab