# Class 5: Symmetric key crypto v1

Note Title

[demo of encryption via addition of a shared secret – transmit a 2-digit "credit card number"]

(1) Basic definitions:

plaintext – the message you want to send
ciphertext – the encrypted version of the plaintext
encrypt – convert plaintext into ciphertext
decrypt – convert ciphertext into plaintext
key – a secret value that is used as an input to the encryption and/or decryption

symmetric key cryptography – uses the same key for encryption and decryption.
(c.f. public key crypto, in next class)

A commonly-used operation is XOR (exclusive or, often written $\oplus$) which combines binary sequences just like addition, except that $1+1=0$ (not 2).

example:

$$\begin{array}{r} 101001 \\ \oplus\ 011010 \\ \hline 110011 \end{array}$$

To undo an XOR, just do it again!

exercise: undo the above XOR operation (i.e. recover the input from the output).

Note: Recall that computers always work with 1s and 0s (binary). It's easy to convert characters to binary — see the link on resource page. e.g. "A" is 01000001. Most of our examples will use binary, but we could easily convert to characters if desired.

## ② One-time pad

A one-time pad encrypts by XORing the message with the key:

$$\text{ciphertext} = \text{plaintext} \oplus \text{key}$$

e.g. Investor wants to send "S" (for 'sell') or "B" (for 'buy').
In binary, S is 01010011
B is 01000010

If the key is 10010101, and the investor wants to sell, what bits should be sent to the stock broker?

→ fill in answer yourself!

Note: To be secure, ① key must be as long as the message
② need a new key for each message.

Why? In investor example above, what happens if they reuse the same key every day? We can observe what actually happens on one day (did they buy or sell?), then the code is cracked for every subsequent day!

[discuss origin of the term 'one-time pad']

So, one time pad only practical if both parties have access to the same, extremely long, random string. This can be done, but there are better ways.

## (3) Block ciphers

A block cipher breaks the message up into chunks called blocks and encrypts each one separately.

e.g. if block size is 128 bits, then an encrypted text message of 48 characters would be broken up into ___ blocks of ___ characters each.

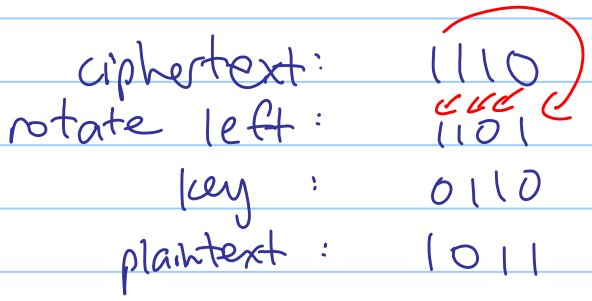
(fill in as exercise)

As a simple example, we use a made-up block cipher called XR (for Xor-Rotate).
XR has a 4-bit block size and a 4-bit key.
To encrypt, we XOR plaintext with the key then rotate the bits to the right by 1 slot:

e.g.

| | |
|---|---|
| plaintext : | 1011 |
| key : | 0110 |
| plaintext ⊕ key : | 1101 |
| rotate right : | 1110 |

To decrypt, rotate to left then XOR with key:




← matches the plaintext we started with, as expected.

Exercise: with the key 1001,

a) encrypt 1111 b) decrypt 0010

## Some famous block ciphers

(a) DES
- 64-bit block
- 56-bit key
- published 1977
- now considered insecure (see Wikipedia page on "EFF DES cracker" – a machine built for $250K in 1998 that can crack DES by brute force within 2-3 days).

(b) AES

- 128-bit block size
- various options for key size, including 128 bits
- published 1998
- considered secure; widely used.

Exercise: Using link on resources page, encrypt "hello" using the key "45654" with AES.

called a "password" on the website.

Now decrypt the result. Do you get back what you started with?

## (4) Cipher-block chaining

Problem with sending a long message via block cipher:
- same input gives same output (when same key is used)
- so we have the same problem as before in the investor/stockbroker scenario.

To fix, use cipher-block chaining:

1. Sender transmits a random initialization vector (IV), which is the same length as the block size, and is sent in the clear (i.e. unencrypted)

2. First block is encrypted by first XORing plaintext with IV, then encrypting.

3. For every subsequent block, first XOR the plaintext block with the previous ciphertext block, then encrypt.

To decrypt, just reverse the above operations.

Example Let's represent "sell" as 0 and "buy" as 1. The investor has twelve stocks and sends instructions for all of them in a single message:

1001 1001 1110

We encrypt with XR, using key 1010 and IV 0110.

1. Transmit IV: 0110
2. Encrypt first block:

| | |
|---|---|
| plaintext: | 1001 |
| IV: | 0110 |
| xor: | 1111 |
| key: | 1010 |
| xor: | 0101 |
| rotate: | 1010 |

→ transmit 1010

3. Encrypt next block:

| | |
|---|---|
| plaintext: | 1001 |
| prev ciphertext: | 1010 |
| xor: | 0011 |
| key: | 1010 |

XOR: 1001

rotate: 1100

→ transmit 1100 (different ciphertext even though plaintext same as block 1 !!)

4. Encrypt next block: complete as exercise.

Decryption:

1. Receive IV: 0011.
2. Receive first ciphertext block: 1010

rotate left: 0101

key: 1010

XOR: 1111

IV: 0110

XOR: 1001

→ first decrypted block is 1001

3. Receive next ciphertext block: 1100

rotate left: 1001

key: 1010

XOR: 0011

prev ciphertext: 1010

XOR: 1001

→ 2nd decrypted block is 1001

4. Do next block as exercise.