CS 455 – Computer Security Fundamentals

Dr. Chen-Yeou (Charles) Yu

Cryptography All-in-One, a full introduction

- ~~Why is Cryptography essential?~~
- ~~What is Cryptography~~
  - ~~How does cryptography work?~~
- ~~Applications of Cryptography~~
- ~~Types of Encryption in Cryptography~~
  - ~~Symmetric Key Cryptography~~
  - ~~Asymmetric Key Cryptography~~
- Hashing
- Data Encryption Standard (DES) Algorithm
- Advanced Encryption Standard (AES) Algorithm
- Digital Signature

Cryptography All-in-One, a full introduction

- Digital Signatures Algorithm (DSA) (TBD)
- Rivest-Shamir-Adleman Encryption (RSA) (TBD)
- Message Digest – 5 (MD5) Algorithm (TBD)
- Secure Hash Algorithm (SHA) (TBD)
- Secure Socket Layer (SSL) Handshake (TBD)
- Diffie-Hellman Key Exchange (TBD)

Hashing

- A Hash function is a set of mathematical calculations operated on 2 blocks of data
- The main raw input is broken down into 2 blocks of similar size. The block size is dependent on the algorithm that is being used
- Hash functions are designed to be one way (irreversible)
- The hashing function can be carried out for multiple times, but the final digest must be consistent for the same input.
- Digest size? It depends on the respective algorithm being used.
- The digest must be the same from the same input

Hash Algorithm		Digest Size
MD5	→	128 bits
SHA-256	→	256 bits

Hashing

- Performance evaluations:
  - There are several different kind of hash functions
  - You can design your own hash function
  - How to evaluate a hash function, good or bad?
  - Send the hashed output for brute force attacks
    - Variable length permutations on a mix of input strings, 0-9, A-Z, a-z, special characters
  - See who can survive in the testing?

Hashing

- Implementation guidelines
  - Inside of the hash function, hash digest must be dependent on each bit of input
  - If a single character is being changed, it doesn't matter how small that character could be, the entire digest must have a distinctly different hash value
    - For example, 2 plaintext(s), "Cryptography" and "Cryptograph", their hashed digest will be totally different!

Hashing

- **There is a very famous problem --- Hash Collision**
  - **You might ask me what if the 2 users are using the same password, after a hash? Their hashed digest are totally the same?**
  - **Unfortunately, the answer is YES! If we keep using the same hashing function.**
  - **There is a thing called “salting”. This can prevent the collisions. We will learn that later**
    - **Want to add some seasoning, i.e. MSG into our dishes? ☺**

Hashing

- Salting
  - It is a process of adding a random keyword to the end of the input, before it is passed to the hash function
  - The random keyword is unique for each user in the system
    - This kind of random keyword is called salt value or just the salt
  - So even if the 2 passwords are exactly the same, if the salt value are different, it still produce different digests
    - 1st one: Just the “Cryptography”
    - 2nd one: with “abc123”
    - 3rd one: with “xyz456”

The hash function is MD5

Input into Hash Function	MD5 Hash Value/Digest
Cryptography	d2fc0657a64a3291826136c7712abbe7
Cryptographyabc123	c56db83ab5482b4e94536f4a29b21de0
Cryptographyxyz456	783b10b483435e05f3f2705bdd5a825c

Hashing

- The idea is good but there is a small problem. The keyword has to be stored in the database along with the original password (We can say it is plain text, before the hashing)
- So, for example, in the previous page, the 2nd row, “Cryptography” has to be with “abc123”
- How to design a database (or tables) to make them connected?
  - i.e. “Cryptography” and “abc123”
- What if the part of salt values in the database is corrupted? It doesn’t necessarily to be hackers doing this. It could be bad blocks from hard drive platters.
- The only one thing we can do is to use “Cryptography”+ variable length string to brute force the hashed digest
  - Otherwise, sender and receiver’s hash doesn’t match. Our web applications will get into trouble.

Hashing

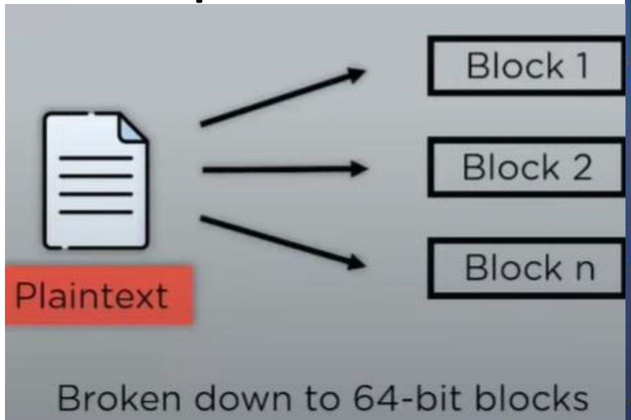
- There is another approach called “Peppering”
  - A lazy-man’s approach to deal with collision
  - If I can add the same “123” to the end of each string, I can make it!
  - It is easy for implementation, but is not secure enough.

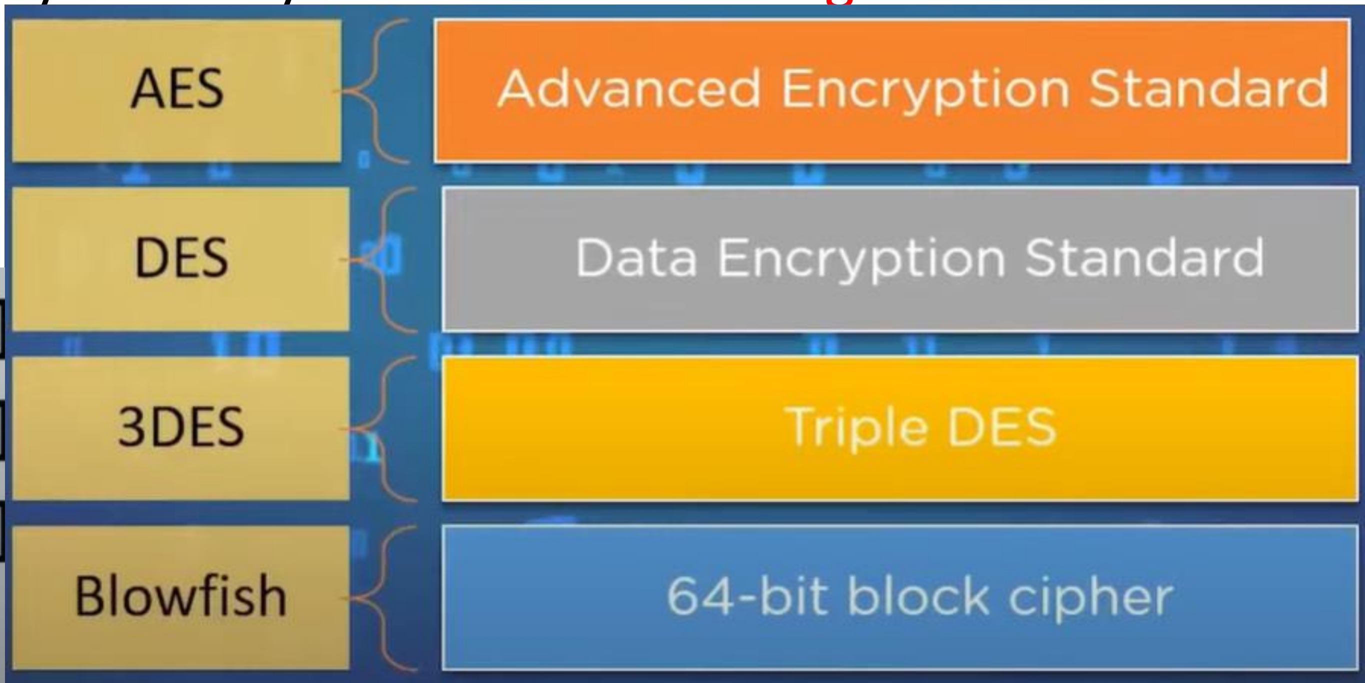
Input into Hash Function	MD5 Hash Value/Digest
CryptographyRan123	4cac3f25fffad414e834ee8208f65116
MyPasswordRan123	470dd61e2acce64486e784f3a288d82f
Qwerty101Ran123	a8792a61ee831c39548ec1a2e1ba3d68

Data Encryption Standard (DES) Algorithm

- Here is the most commonly seen family of Symmetric Encryptions
- Unlike the pure hashing, **we need to use key now!**
- It is block-by-block encryption like we introduced earlier
  - Each block is encrypted individually and they are later chained together to form our final cipher text
  - Then? Transfer the data!
- Plaintext: 64 bits per block
- Key size:

48 bits



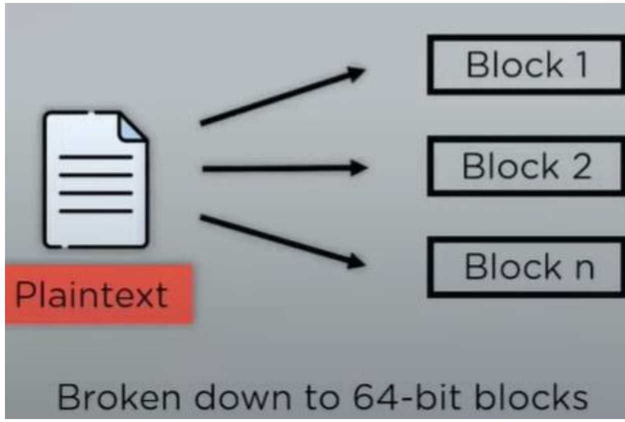


Data Encryption Standard (DES) Algorithm

- Evolution of DES?
  - DES is easily cracked now and 3DES is introduced (symmetric)
  - 3DES. Very Slow. They use 3 keys
    - Encrypt with the 1st key.
    - Delete encryption with 2nd key
    - Encrypt again with the 3rd key
  - It is no more useful for the needs of fast communication.
  - It is eventually ended in 2002, replaced by AES (Advanced Encryption Standard)

Data Encryption Standard (DES) Algorithm

- Let's take a closer look inside of this DES algorithm
- We really like to know what happens or how does each of the block being processed?
    - i.e. Block1
- DES makes use of the idea of Feistel Cipher as its Internal encryption/decryption algorithm



Data Encryption Standard (DES) Algorithm

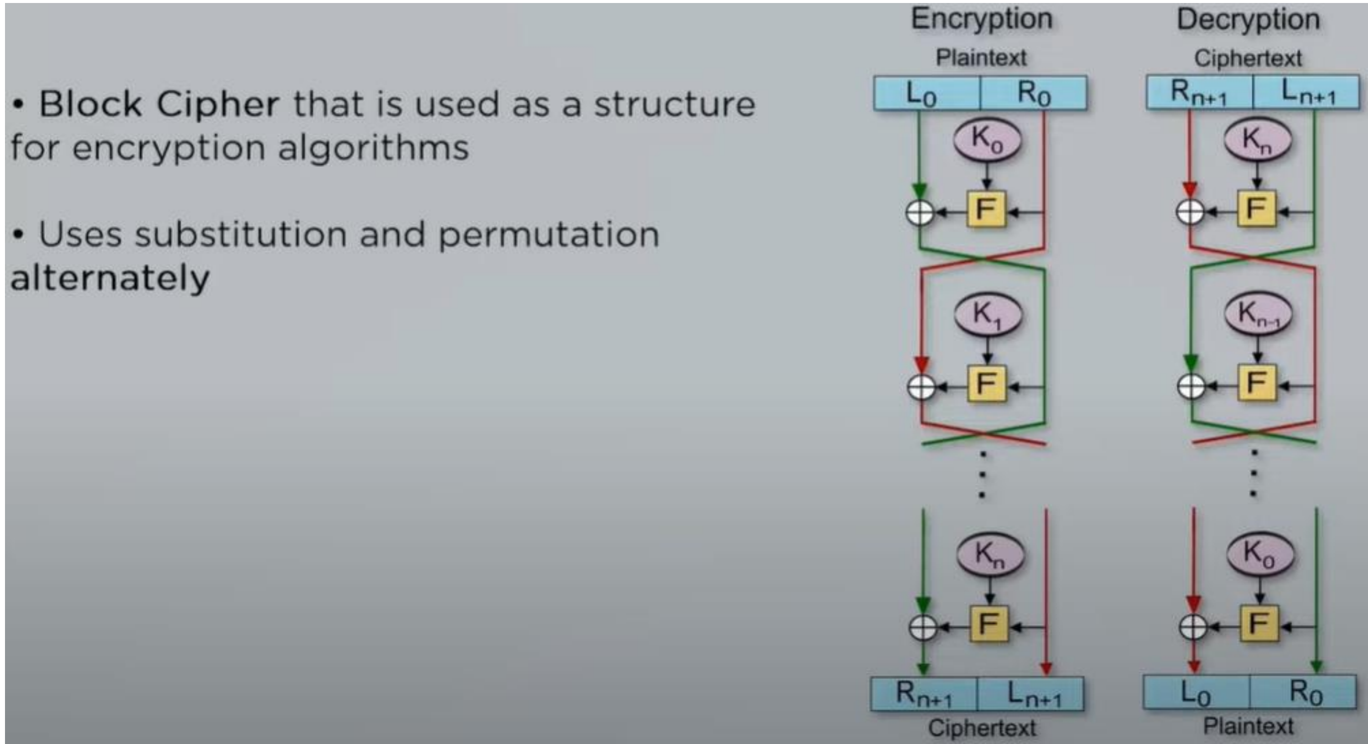
- Oh! Wow! For a single “Block1” it has to experience these complicated processing
- Tear the Block1 into 2 pieces L_0 and R_0
- Keys are sliced into “n” sub-keys
- There are totally “n” rounds

of processing

- F is the round function
- “+” sign is the XOR
- After the encryption, the original

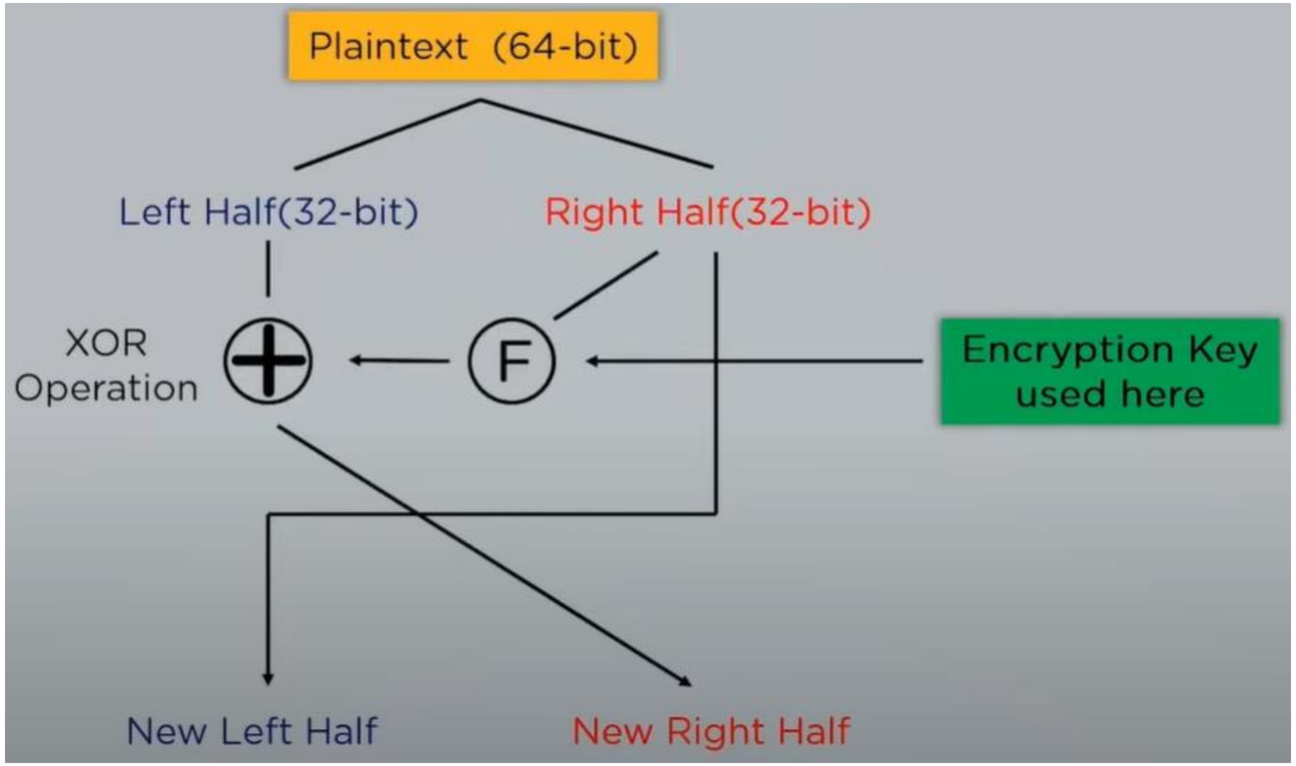
LHS and RHS is swapped

Eventually, it will be swapped back!



Data Encryption Standard (DES) Algorithm

- The single round is shown here! (Encryption)
- But are you curious what happens inside of F?

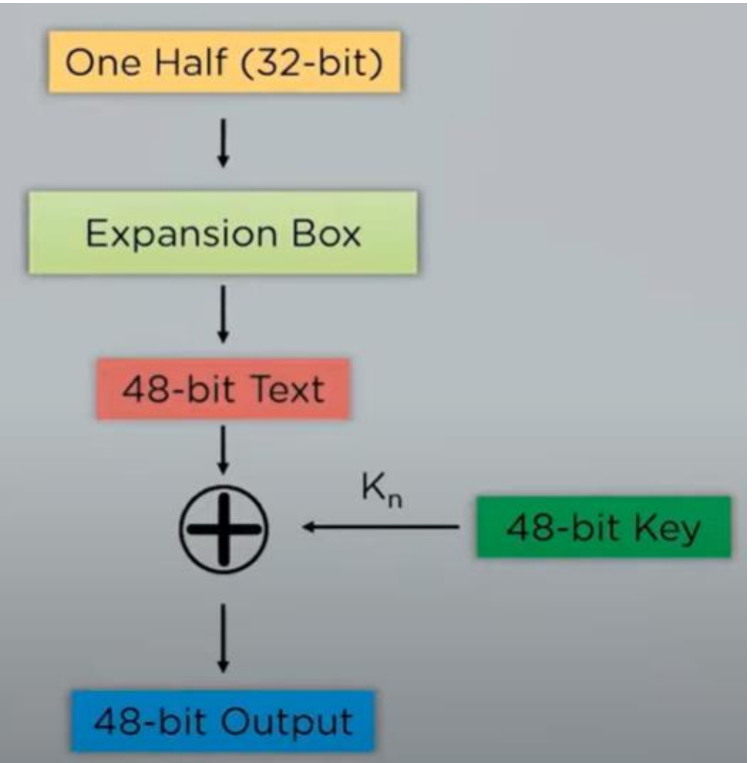


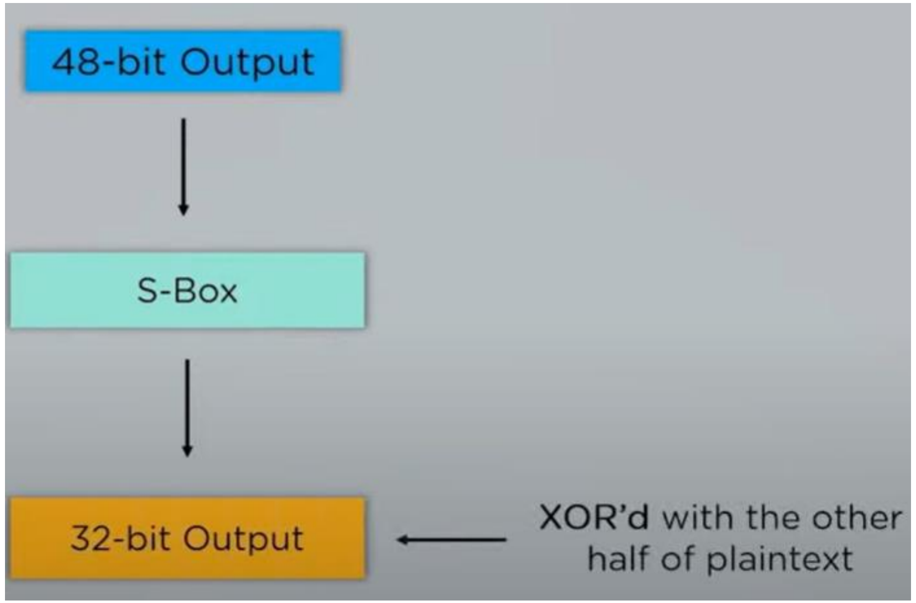
Data Encryption Standard (DES) Algorithm

- Let’s take a deeper look at what’s happening in the “Right Half”
- F is so-called round-function
- What happens inside of the F?
  - Inside of the round-function, there will be an XOR computation
  - So there are totally 2 XOR in a round. One is inside of the F and another is outside of F
  - The job of expansion box is to increase the size of the half from 32 bit to 48 bit. (check the next page)
  - So in this way we can use XOR computation with our 48 bits key

Data Encryption Standard (DES) Algorithm

- This is our “round” function
- S-Box is the substitution box





Data Encryption Standard (DES) Algorithm

- Implementation guidelines

(maintaining the speed and security is a kind of philisophy)

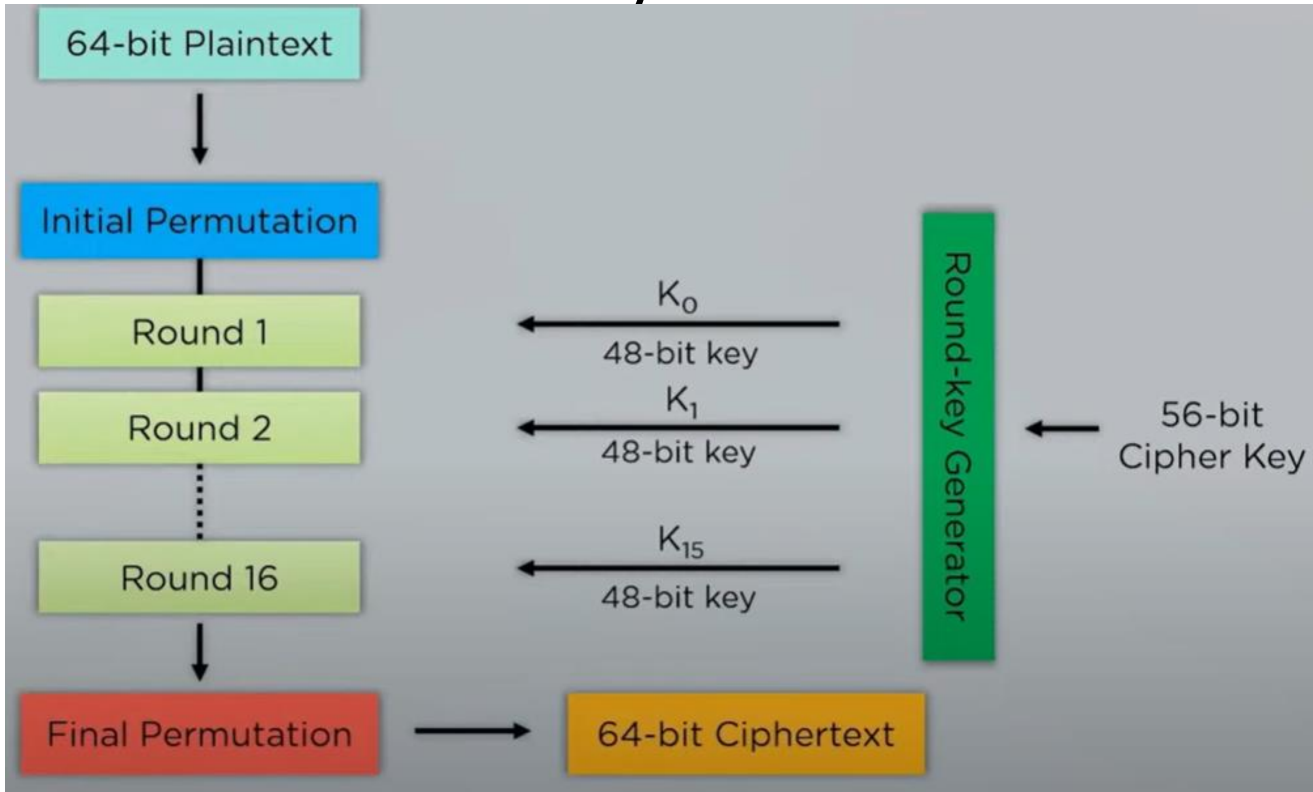
  - Generally, block size is 64 bits but it should be a compromise between size and speed (if we can choose)
  - Key size is directly proportional to strength of encryption.
  - Larger key size reduces the speed
  - Higher number of rounds is getting harder to crack

Data Encryption Standard (DES) Algorithm

- The encryption process! (16 rounds)
  - Initial Permutation is to 1) permutation 2) slice it into 2*32-bits blocks
- The Round-key generator will generate 16 different keys for each of the single round
- For the final permutation,

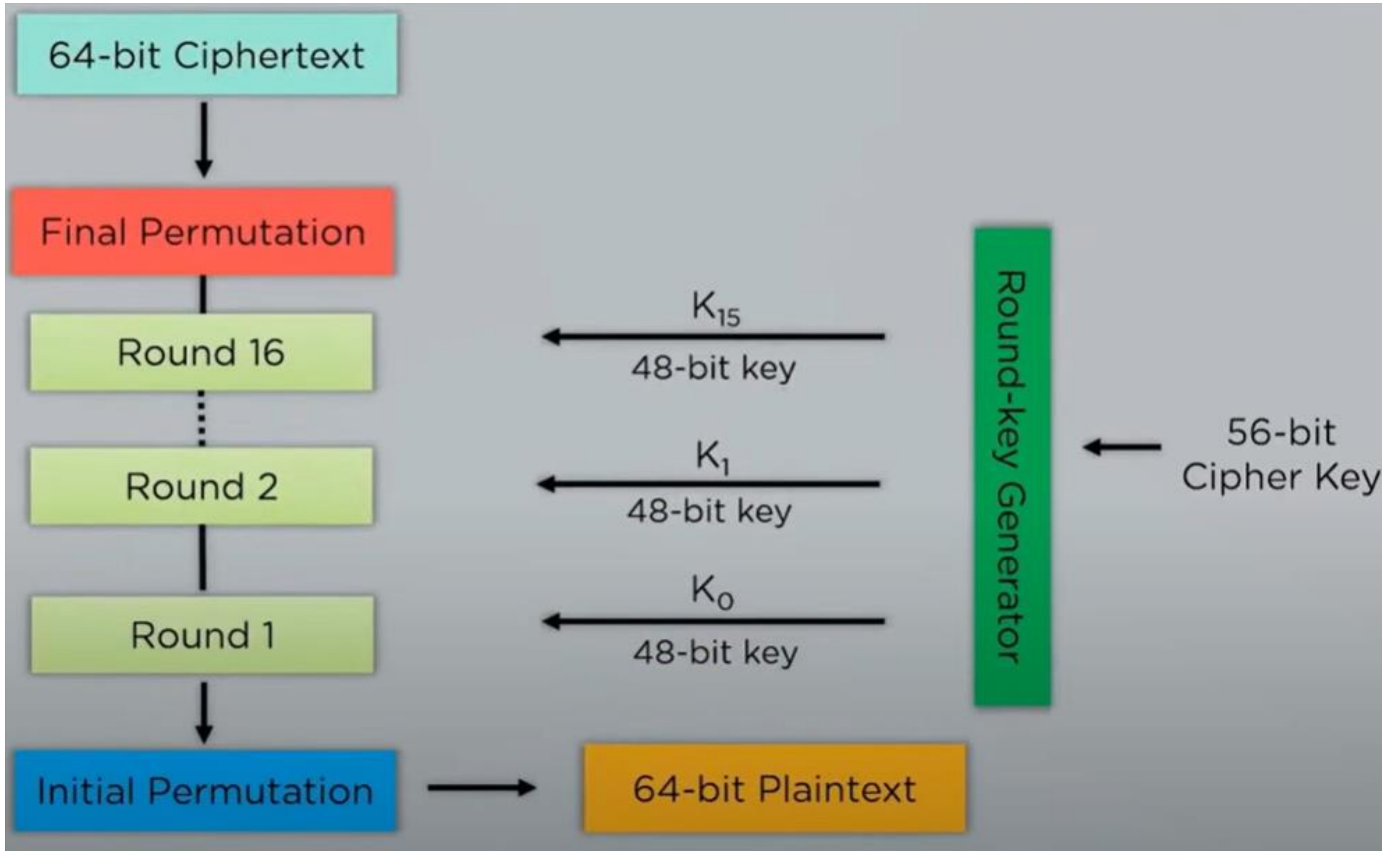
the 2 parts are swapped back and

we get the 64bit cipher text



Data Encryption Standard (DES) Algorithm

- The decryption process! (16 rounds)



Advanced Encryption Standard (AES) Algorithm

- DES is totally dead!
- AES?
  - Symmetric key
  - Block based
  - Each of the block size is 128 bits = 16 bytes
  - Key is in the size of 128, 192 or 256 bits
  - It is, the same, uses substitution and permutations like DES
  - AES performs on the byte data, instead of the bit data (performed by DES)
  - The number of rounds of computation depends on key length

Chronology of DES Cracking	
Broken for the first time	1997
Broken in 56 hours	1998
Broken in 22 hours and 15 minutes	1999
Capable of broken in 5 minutes	2021

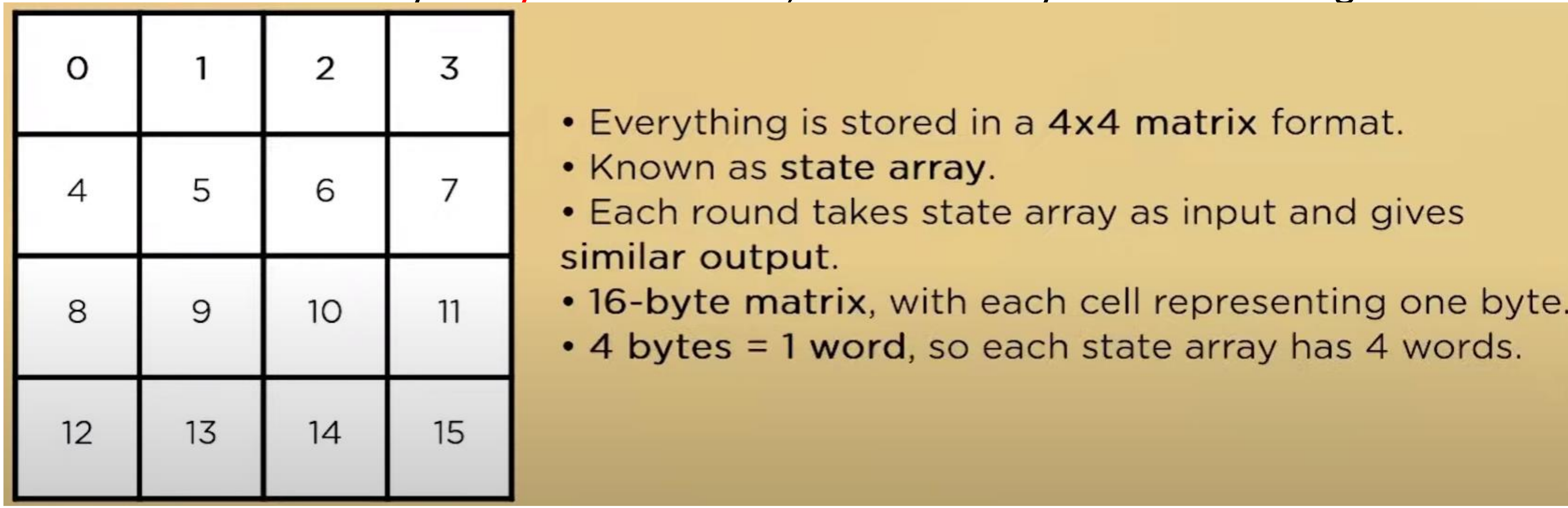
128-bit Key Length → 10 rounds

192-bit Key Length → 12 rounds

256-bit Key Length → 14 rounds

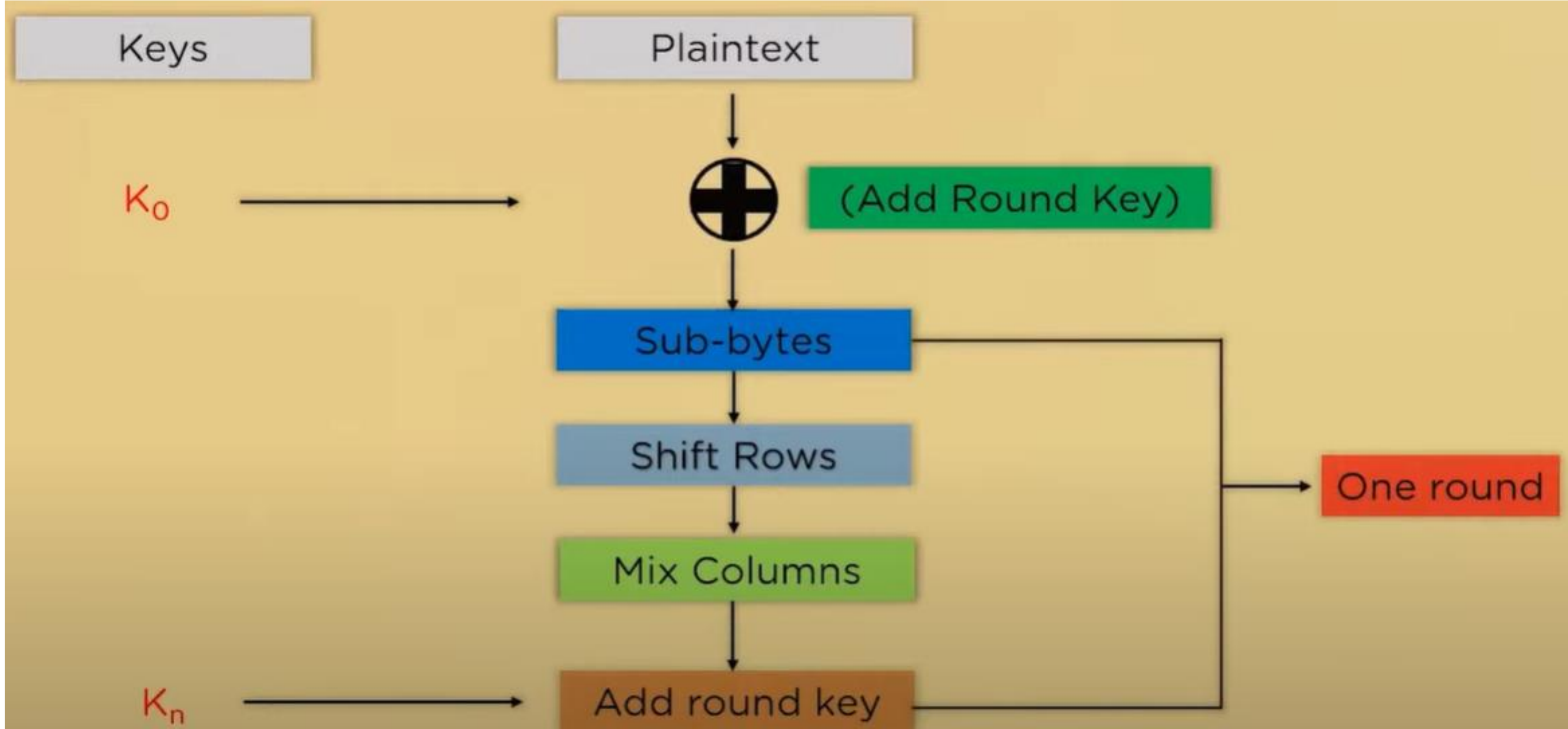
Advanced Encryption Standard (AES) Algorithm

- The manner of storage
  - Since there are totally 16 bytes for a block, here is the layout of the storage



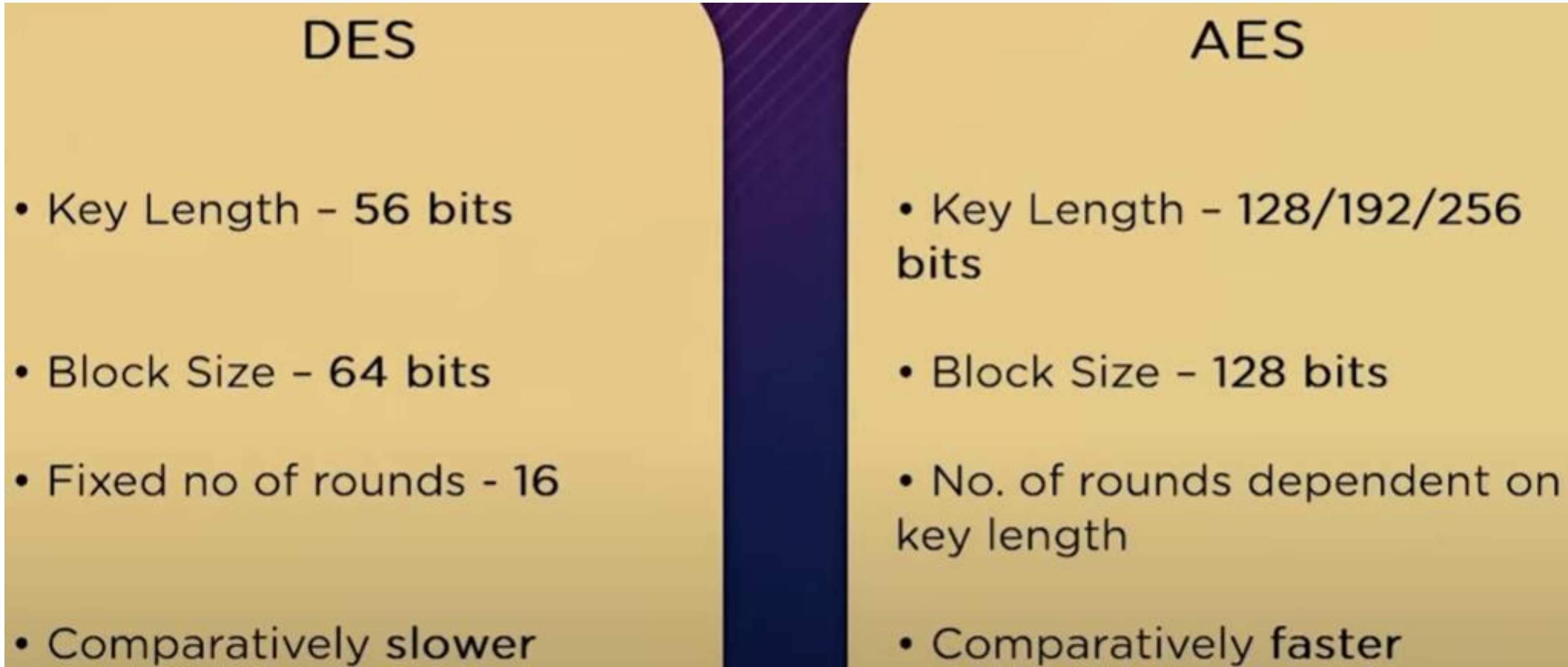
Advanced Encryption Standard (AES) Algorithm

- It is similar to DES, the “round” based encryption.
- The process of encryption is overly complicated and is beyond of the scope of this class. For details, we just skip it.



Advanced Encryption Standard (AES) Algorithm

- Any Applications?
  - Yes! It is SUPER widely used in WiFi password encryptions

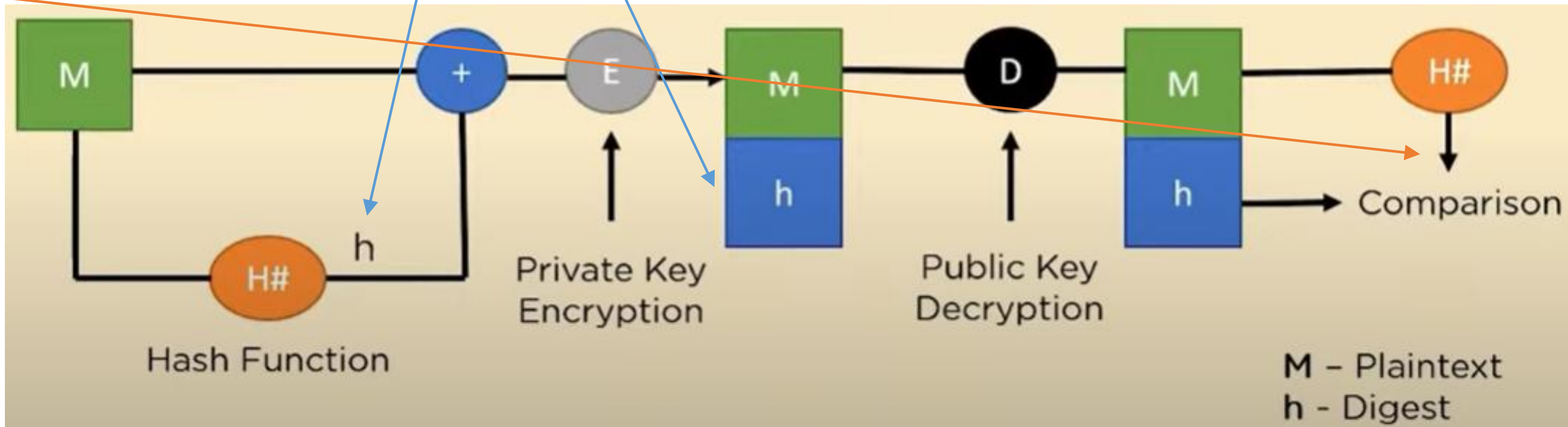


Digital Signature

- Typically, for an asymmetric key system, the encryption, it uses
  - public key
- For decryption, it uses
  - private key
- However, I need to put a comment here! For signing (digital signature), it is reversed!
  - Encryption is using private key
  - The signature is verified (decrypted) by using public key

Digital Signature

- Here is the detail process
  - M: original plain text message
  - H#: the hash function where M is passed into
  - h: hash digest created. We can say, this is h1
  - M and h are bundled together and is encrypted by using sender’s private key “E”
  - Once the message “M” is decrypted, it is passed to the same hash function H# to generate a digest h2
  - If h2==h1, it verifies the data integrity in “M”



Digital Signature

- Next, the Digital Signature implementation has 2 majority of algorithms.
- We will talk about that in the next Monday