Topic 2

Stream Cipher

- Stream ciphers encrypt individual bit or character streams
    - When encrypting individual bitstreams, XOR is used, i.e.
        - ciphertext (C) = plaintext (M) xor keystream (KS)
    - Replace the random key in One-time Pad with a pseudo-random sequence

Topic 3 Symmetrical cryptography

Block Cipher

- Definition: Plaintext is divided into blocks and encrypted one at a time. If last part of the message is shorter than the block size - add padding
- Requirements:
    - Completeness
    - Avalanche Effect (Diffusion) 雪崩效应/扩散
    - Statistical independence (Confusion) 统计独立性/混淆
- Difference between block cipher and stream cipher
    - BLOCK CIPHER Uses both confusion and diffusion,
      STREAM CIPHER Relies on confusion only
    - Algorithm modes used:
        - BLOCK CIPHER: ECB (Electronic Code Book) & CBC (Cipher Block Chaining)
        - STREAM CIPHER: CFB (Cipher Feedback) & OFB (Output Feedback)
    - Reversibility:
        - BLOCK CIPHER: hard
        - STREAM CIPHER: It uses XOR for the encryption which can be easily reversed to the plain text.

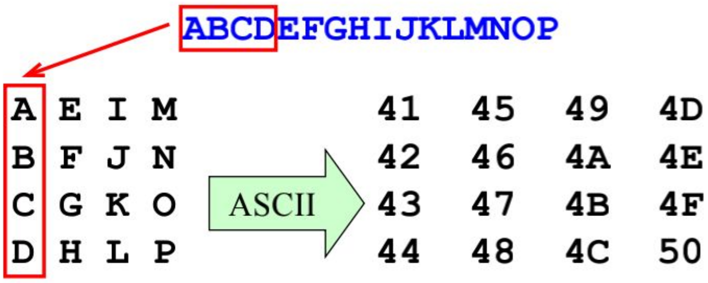
Feistel Block Cipher

DES (Data Encryption Standard)

AES (Advanced Encryption Standard)

- AES is a symmetric block cipher, but it is an iterative
    - Encryption and Decryption using the same key
    - Plaintext and ciphertext have the same size

    - It is a substitution-permutation cipher involving r rounds
    - Standard:
        - Block size: 128 bits/16 bytes, called state



        - Key length: 128(r=10), 192(r=12), 256(r=14)
- Round transformation
    - Substitute bytes (SubBytes) -- 1 S-Box
    - Shift rows (ShiftRows) -- do (n-1) left shift
    - Mix columns (MixColumns).
    - Add round key (AddRoundKey).

Modes of Encryptions

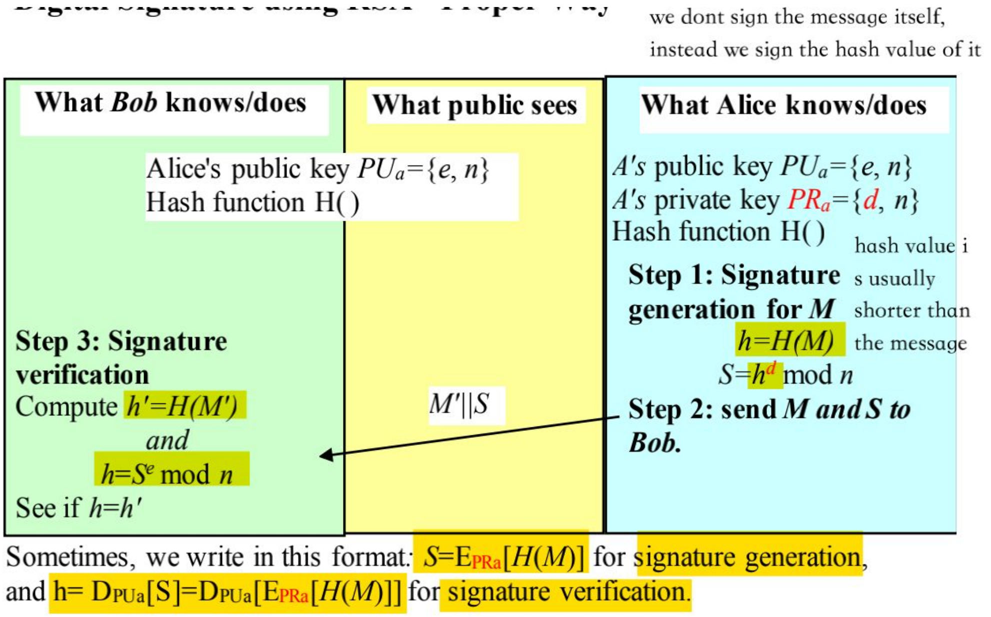
- ECB – Electronic Code Book mode
- CBC – Cipher Block Chaining mode
- CTR – Counter mode

Topic 4 Public-Key Cryptography(PKC)
RSA, DSS, DH

RSA

- It's a block cipher. Block length is limited by the modulus n. The plaintext and ciphertext, when converted to integers should be between 0 and n-1
- Key Generation:
    - Select two large prime, p and q [SCRECT]
    - n = p*q, φ(n) = (p-1)(q-1) [PUBLIC]
    - Select int e, which is relatively prime to φ(n) & 1<e< φ(n) [PUBLIC]
        - Relatively prime: a and b are relatively prime iff gcd(a,b) = 1
    - d = e-1 mod φ(n) (or d.e = 1 mod φ(n)) [PRIVATE]
        - Security of RSA relies on the difficulty of finding d, given {e, n}
    - So we got:
        - public key = {e, n}
        - private key ={d, n}
- Encryption:
    - Plaintext M, (M<n)

  - Ciphertext C = M^e mod n (NOTE: not φ(n))
- Decryption:
  - Plaintext M = C^d mod n
- RSA vs AES
- RSA vs DSA
- RSA can also be used in digital signature



we dont sign the message itself, instead we sign the hash value of it

What *Bob* knows/does	What public sees	What Alice knows/does
Alice's public key $PU_a=\{e, n\}$ Hash function H( )		$A's$ public key $PU_a=\{e, n\}$ $A's$ private key $PR_a=\{d, n\}$ Hash function H( )
		Step 1: Signature generation for *M* $h=H(M)$ $S=h^d \bmod n$
	$M'\|\|S$	**Step 2: send *M and S to Bob.***
Step 3: Signature verification Compute $h'=H(M')$ *and* $h=S^e \bmod n$ See if $h=h'$		

hash value is usually shorter than the message

Sometimes, we write in this format: $S=E_{PRa}[H(M)]$ for signature generation, and h= $D_{PUa}[S]=D_{PUa}[E_{PRa}[H(M)]]$ for signature verification.

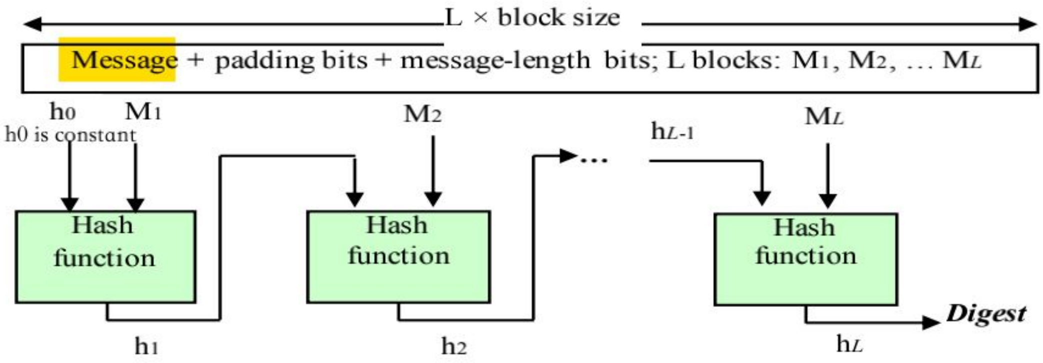
  - All possible ways of forging a signature.
    - Use a signature signing key (private key) with a decent length.
    - Use a secure/strong hash function.
    - Timing sources tamper-proof and synchronised, or include a random number
    - (nonce) contributed by the verifier in the signature.
    - Obtain someone else's private signature key
      - In a digital signature scheme, "you are your private key".
    - Persuade others that someone else's public verification key belongs to you.
  -
-

Topic 5 Cryptographic Checksums

Terminology

- Checksum = digest = fingerprint = compressed value = hash value MAC (Message Authentication Code) = cryptographic checksum

Hash Function

- Overview:
    - Given a message M of arbitrary length, produces a fixed-length output
    - A Many-to-one mapping so collisions are unavoidable: if the output is shorter than its input then there is a possibility that two different messages may hash to the same value
- Compression Property
    - if the message to be hashed is longer than the block size,u need to div the message into many blocks for the last u need padding
    - 

h0 is a constant initial value, and the output of the last block *is the digest* of the entire message.

- Security Requirements/Properties
    - Preimage resistant (one-way)
        - Given a hash value H_M it is infeasible to find the message M that hashed to that value h(M) = H_M
    - 2nd preimage resistant (weak collision-resistant) -> selective forgery
        - Given a message it M and H(M) it is computationally infeasible to find another M' s.t. H(M') = H(M)
        - The difference from strong collision-resistant: it is for a given M
        - Signature forgery if weak collision resistance property is not met.
    - Collision resistant (strong collision-resistant) -> existential forgery
        - It is computationally infeasible to find any M and M', M ≠ M' with H(M) = H(M')
        - If it is strong collision-resistant, then it is also weak collision-resistant
        - Repudiation if strong collision resistance property is not met.

Block Cipher based MAC (Message Authentication Code)

- Overview(from wiki)
    - A cryptographic checksum on data that uses a session key to detect both accidental and intentional modifications of the data. 是对数据的一种加密校验，它使用会话密钥来检测数据的意外和故意修改。
    - A MAC requires two inputs: a message and a secret key known only to the originator of the message and its intended recipient(s). 一个MAC需要两个输入：一个电文和一个只有电文发端人及其预定接收人知道的秘密密钥

        - This allows the recipient of the message to verify the integrity of the message and authenticate that the message's sender has the shared secret key. 这使得信息的接收者可以验证信息的完整性，并验证信息的发送者是否拥有共享的秘密密钥。
- MAC = f_k (M), f_k is a block cipher based digest function that has an embedded key.
- Security:
    - MAC achieves message authentication (origin + integrity)
    - A nonce could be used for added freshness (anti-replay)
- MAC is NOT a digital signature. No non-repudiation

Authenticated encryption

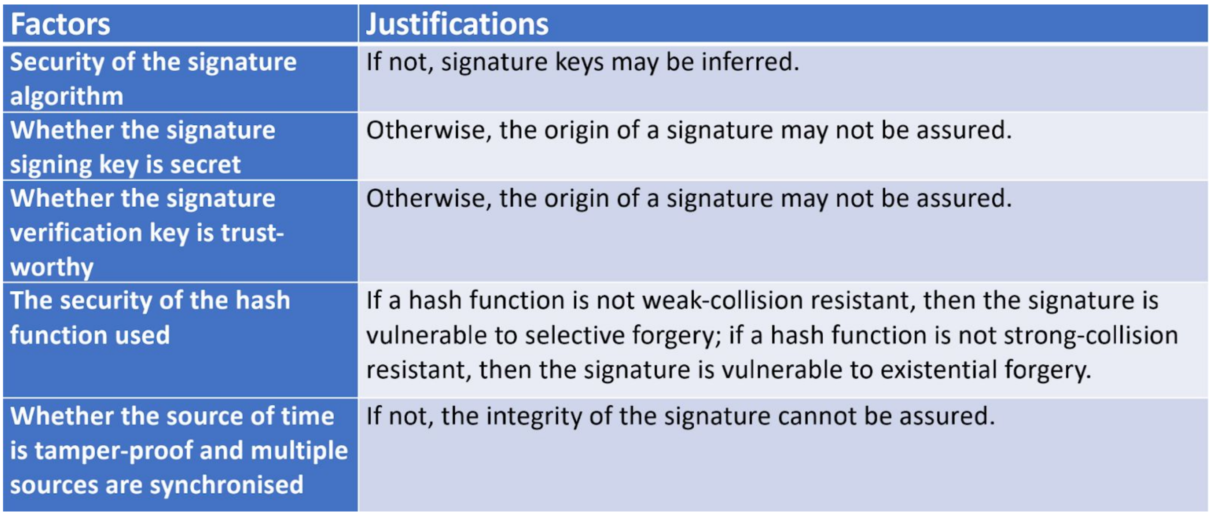
- General Concept:
    - MAC(CBC and HMAC) - message authentication (origin + integrity)
    - Authenticated encryption = message authentication + confidentiality
    - Possible approaches:
        - Hash-then-encrypt: E(K, M||H(M))
            - to produce the checksum,bcz it does not has an embedded key so in this case, the hash value is encrypted along with the message
            - Runtime: hash operation + the block cipher operation
        - MAC-then-encrypt (used in SSL): E(K2, M||MAC(K1, M))
            - MAC(K1, M) - for origin authentication
            - is more vulnerable to DoS attacks since the recipient needs to perform 2 operations before discarding a message. First, decrypt and only then verify the MAC
            - Runtime: 2 * block cipher operations
        - Encrypt-then-MAC (used in IPSec): MAC(K1, E(K2, M))
            - the recipient can discard a message without decrypting it. Verify the MAC and only then decrypt it if the MAC is okay.
            - Encrypt-then-MAC has a higher throughput than MAC-then-encrypt this is why it's used in IPSec
            - Runtime: 2 * block cipher operations
        - Encrypt-and-MAC (used in SSH): E(K2, M)||MAC(K1, M)
            - quickest as it is parallelizable
            - Runtime: one block cipher operations

Topic 6 Digital Signature

General Concept

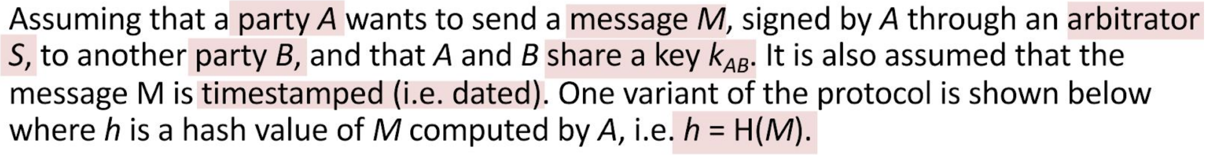
- Security Requirements:
    - message-dependent, inc date/time
        - unreusable
        - ensures content integrity

    - signer-dependent
        - unforgeable
        - ensures origin authentication
    - verifiable: others should be able to verify the validity of a signature.
    - anti-forgery: computationally infeasible to forge.
- Forgery types:
    - Existential forgery
    - Selective forgery
- Security Factors:

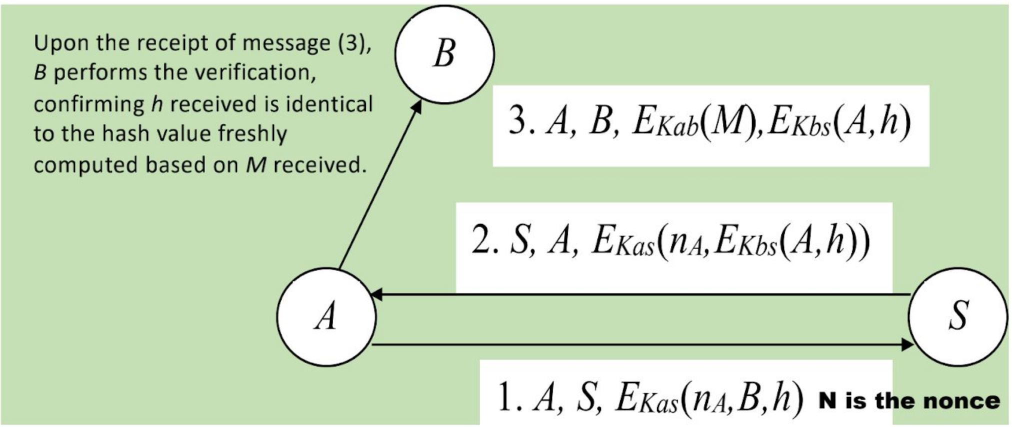


Factors	Justifications
Security of the signature algorithm	If not, signature keys may be inferred.
Whether the signature signing key is secret	Otherwise, the origin of a signature may not be assured.
Whether the signature verification key is trust-worthy	Otherwise, the origin of a signature may not be assured.
The security of the hash function used	If a hash function is not weak-collision resistant, then the signature is vulnerable to selective forgery; if a hash function is not strong-collision resistant, then the signature is vulnerable to existential forgery.
Whether the source of time is tamper-proof and multiple sources are synchronised	If not, the integrity of the signature cannot be assured.

Digital Signature using Symmetrical Key (Needs a trusted third party)

- 

Assuming that a party *A* wants to send a message *M*, signed by *A* through an arbitrator *S*, to another party *B*, and that *A* and *B* share a key k_{AB}. It is also assumed that the message M is timestamped (i.e. dated). One variant of the protocol is shown below where *h* is a hash value of *M* computed by *A*, i.e. *h* = H(*M*).



Upon the receipt of message (3), *B* performs the verification, confirming *h* received is identical to the hash value freshly computed based on *M* received.

B
3. $A, B, E_{Kab}(M), E_{Kbs}(A,h)$
2. $S, A, E_{Kas}(n_A, E_{Kbs}(A,h))$
A
S
1. $A, S, E_{Kas}(n_A, B, h)$
N is the nonce

- Different between using Symmetric key and RSA

    - The RSA signature scheme only requires an off-line trusted third party (TTP), whereas this one requires an on-line TTP.
    - The RSA scheme does not require a shared secret, rather the signer needs to have a key pair, and the signature verification key must be certified by a trustworthy CA, whereas the above signature protocol requires a method for symmetric key distribution.
    - With the RSA scheme, the signer experiences more computational cost, but less communicational costs, than the symmetric scheme.

Topic 7 Public Key Infrastructure (PKI)

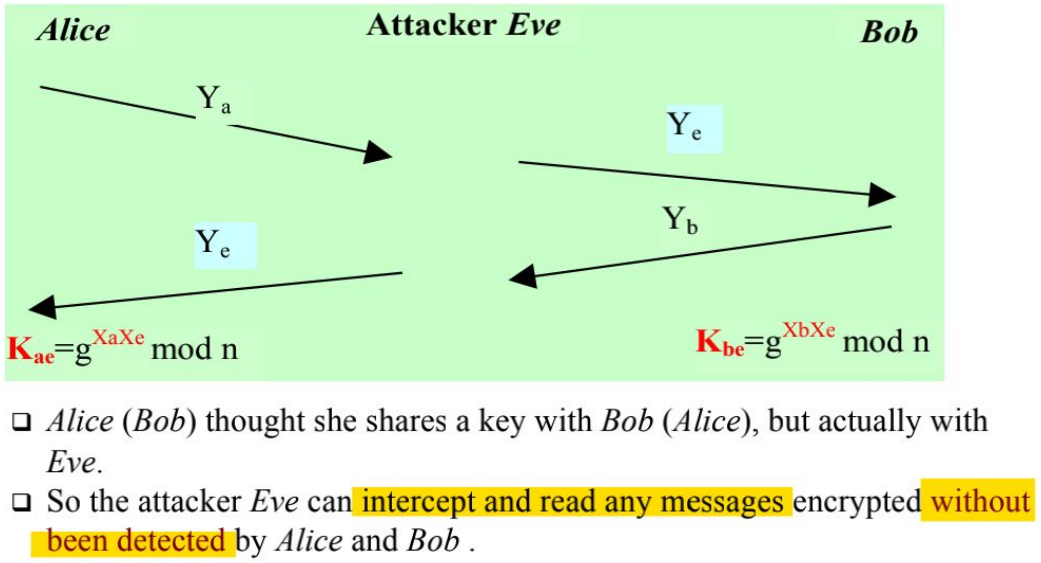
A digital certificate = public-key certificate = digital ID = certificate

- Binds an entity's public key (+ one/more attributes) to its identity (the entity = person, hardware device, software process). It is digitally signed by the CA so you need CAs public key to verify the certificate
- Application:
    - a certificate for secure email contains the entities email address
    - a certificate for the financial purpose may contain a credit card number and credit limit
-

Topic 8 Key Management

DH Protocol/Algorithm (Diffie-Hellman)

- Man-in-Middle Attack
    - This is an active attack. The attacker intercepts and substitutes Ya and Yb with Ye. So at the end of the message exchanges, Eve will have Kae, Kbe, Alice will have Kae and Bob will have Kbe. Any messages encrypted with Kae will be decrypted and read by Eve and then re-encrypted with Kbe and vice versa without the knowledge of Alice and Bob.

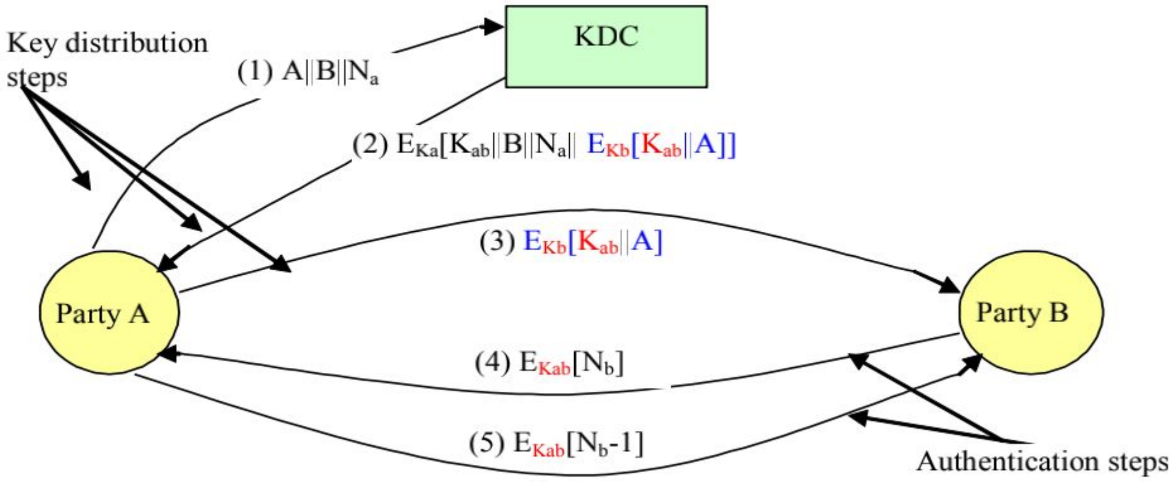


❑ *Alice* (*Bob*) thought she shares a key with *Bob* (*Alice*), but actually with *Eve*.

❑ So the attacker *Eve* can intercept and read any messages encrypted without been detected by *Alice* and *Bob* .

- ○ Solution
    - ■ Fixed DH
    - ■ Ephemeral DH

Needham-Schroeder protocol

- 

- ○ Step (3):
    - ■ Reduced involvement of KDC, thus less overhead not just for KDC also for B;
    - ■ This protocol (designed in this way) can also be used for authentication purpose, in addition to confidential communication between A and B.
- ○ Step (4)(5):
    - ■ Demonstrate A is the origin of the messgae
- ○ Application:
    - ■ Establishing a secure communication channel
    - ■ Authentication service.
    - ■ Benefits:

- Party A (i.e. user) does not need to remember many keys while being able to use a different key for a different correspondent;
- When used for authentication, the protocol supports single sign-on, i.e. a user only need to remember a single password, but is able to use different short-term secrets for different servers and the users' master secrets (i.e. long-term passwords) are only managed by one entity, i.e. KDC.

Topic 9 Entity Authentication

Unix authentication - authN process

- When a user tries to log in, the program /bin/login takes the password the user typed, and the salt from the password file, to generate a fresh encrypted password, and compares the newly generated one with the one stored in the server. If the two encrypted results match, the system lets you in.
    - Login process: E(plaintext_pw||salt, 00000000) == stored_encrypted_pw ?
- Uses Hashed PW with Salt
- The hash function is the Crypt() algorithm
    - based on the DES algorithm
    - 8 character password form 56-bit key (7 bits ASCII + 1 parity)
    - 12-bit salt to
        - perturb the DES algorithm, to prevent dictionary attacks
        - prevent identical PWs from producing the same hashed PW
- The PW is used as the key to the DES algorithm
    - cannot reverse from encrypted password to the plaintext password
    - No need for a separate encryption key. The server doesn't have to store encryption keys for every user.
    - The server doesn't have to store and manage plaintext passwords
- Possible attack and solutions to it:
    - Dictionary Attack: /etc/passwd file needs to be accessed by processes
        - Solution: Put pwd in a shadow password file, /etc/shadow, is used, which contains the encrypted passwords, and is put in a root account; put an x (or other placeholders) in the original /etc/passwd file.
    - Replay Attack: An attacker can eavesdrop on a network to get your login ID and encrypted password and later replays (re-send) it to gain access to the system

        - Solution: overlook this problem in a LAN environment, bcz LAN is secure, eavesdropping can be noticed

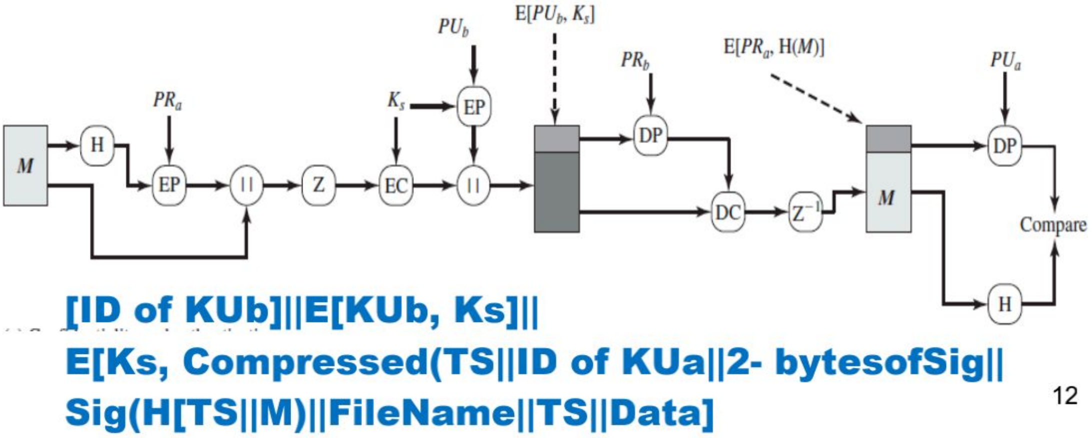
Topic 10 A Security System (PGP)

PGP (Pretty Good Privacy) - Email

- PGP vs X.509
    - The one significant advantage that PGP had, was that no certification infrastructure was needed before anyone could use PGP.
    - Not scalable when a large number of users are involved.
    - Certificates do not have a period of validity (they are infinitely valid), and their revocations are left to the user not by the CA.
    - The certificate structure gives no indication of any authorization associated with key; again, this is left to the users.

PGP Services

- Confidentiality only, Authentication is done by digital signature
    -



[ID of KUb]||E[KUb, Ks]||
E[Ks, Compressed(TS||ID of KUa||2- bytesofSig||
Sig(H[TS||M)||FileName||TS||Data]

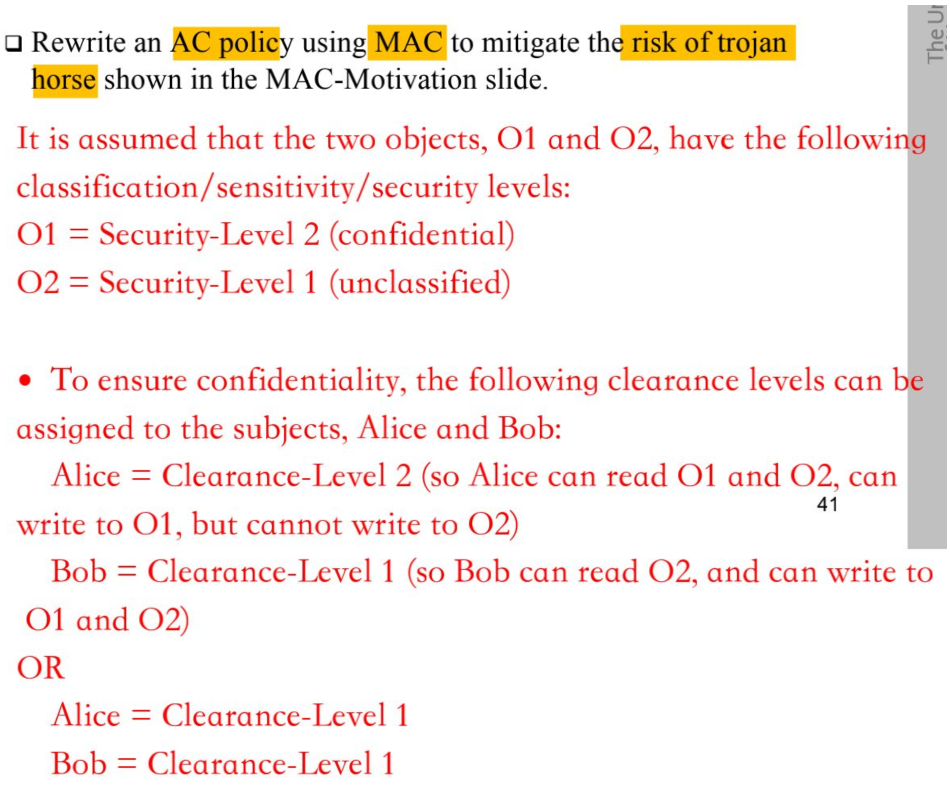
- Compression-then-encryption
    - Benefit: Signature verification can only be carried out after two decryption operations and this makes the solution vulnerable to DoS attacks. However, as this is an end user to end user communication, so the issue of DoS attacks is considered not as important as users’ preference of record- keeping a signed email (on plaintext). Compression-then-encryption can bring two benefits, one is security (harder to break a text that is not recognisible) and the other is performance (cheaper to encrypt a shorter message)
      一个是安全性（更难破解一个无法识别的文本），另一个是性能（加密一个较短的邮件更便宜）。
    - Weakness: This order of operations is not suited to the cases where the operations are carried out by intermediary devices, as this would impacton

the performance–reducing throughput and vulnerableto DoS attacks.
这种操作顺序不适合由中间设备进行操作的情况，因为这会影响性能--降低吞吐量，容易受到DoS攻击。

Topic 11 Access Control

Access Control (AC)

- Discretionary Access Control (DAC)
    - Identity-based access control
    - Directory Access: lists all the files a user (subject) can access
    - Problem:
        - The list can become too large if there are many shared objects;
        - Revocation of access rights can be time-consuming;
        - Pseudonyms or file naming inconsistency;
        - The approach can not efficiently address most object protection scenarios.
    - Access Control Matrix(Capability) & Access Control Lists
    - DAC models are flexible in terms of policy specification, and are typically used by OS and DBMS in commercial world. However, they do not provide information flow control, thus vulnerable to trojan horse attacks.
- Mandatory Access Control (MAC)
    - Classical model: clearance-based access control
    - MAC can protect against Trojan Horse attacks, and is typically used in military applications, and/or for constructing trusted computing systems.
    -



❑ Rewrite an AC policy using MAC to mitigate the risk of trojan horse shown in the MAC-Motivation slide.

It is assumed that the two objects, O1 and O2, have the following classification/sensitivity/security levels:
O1 = Security-Level 2 (confidential)
O2 = Security-Level 1 (unclassified)

- To ensure confidentiality, the following clearance levels can be assigned to the subjects, Alice and Bob:

  Alice = Clearance-Level 2 (so Alice can read O1 and O2, can write to O1, but cannot write to O2)

  Bob = Clearance-Level 1 (so Bob can read O2, and can write to O1 and O2)

OR

  Alice = Clearance-Level 1

  Bob = Clearance-Level 1

- Role Based Access Control (RBAC)
  - A user’s permissions are determined by users’ roles, rather than identity or clearance level.
  - Roles can encode arbitrary attributes.
  - Scalable solution by resembling organizational structures in large organizations.
  -
-
-