CS3235 Part1 Lec3

$_{/}|/|U_Ch@NgRu!$

December 16, 2021

1 Hash Function(cryptographic checksum or msg digests)

1.1 General Hash definition

arbitrary length **message** → **Hash** → fixed sized **digests**

1.2 Security requirements

1. One-way or preimage resistent: given a digest d, it's hard to find a message m such that H(m)=d
2. Second pre-image resistent: given a message m_1, it's hard to find a message m_2, $m_1 \neq m_2$, such that $H(m_1) = H(m_2)$
3. Collision resistent: it's hard to find any m_1, m_2, $m_1 \neq m_2$, such that $H(m_1) = H(m_2)$
4. H(x) should look random, every bit equally likely to be 0 or 1

1.3 Hash VS Encrytption

	Hash	CPA Encryption
One-way	Yes	No
CPA secure	No	Yes

1. Hash is one-way while CPA encryption is not, which means theres is no "De-Hash", even if key is known
2. Hash verify is deterministic, which means we can use H(m) to determine whether m is equal to certain value(also by second preimage resistent)

1.4 Application of hash

1.4.1 Password hashing

1. When user register enter password, compute s=h(password) and store only
2. When user login, enter password', compute h(password') and compare whether s=h(password')

1.4.2 Software integrity

1. If having some secure channel: apply hash
2. If having shared key k: apply mac
3. otherwise: apply signature

1.4.3 Integrity

- Source integrity(Data-Origin Authenticity)
- Data integrity: Unkeyed Hash

Authentic is an adjective to say that the claimed entity/origin is assured by supporting evidence. Authenticity is the condition of being authentic.
Authenticity and integrity are thus related. In the context of an insecure channel, we can say that a message that has been modified in transit means that it no longer comes from its original source.
In other words, a message whose integrity is compromised also means that its authenticity is compromised. As such, data-origin authenticity implies data integrity. But data integrity does not imply data-origin authenticity. Authenticity is thus a stronger requirement than integrity.

2 HMAC

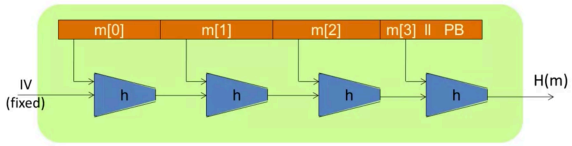
- Constrcut MAC from cryptographic hash function
- Used in SSL/TLS, IPsec (HMAC-SHA-1-96)

2.1 Reason to use HMAC

1. faster than encryption
2. Library for hash widely available
3. can replace one hash function with another
4. There used to be US restriction on encryption

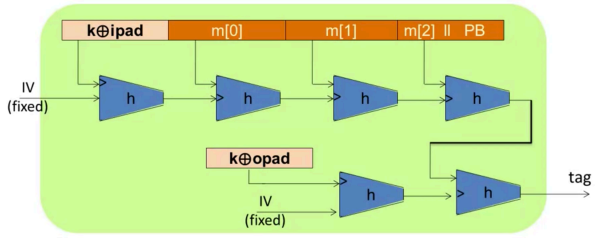
2.2 Construction of HMAC

2.2.1 Construction 1



Given H(k||m) can compute H(k||m||padingBit||w) for any w. In othber words, it's hash extendable

2.2.2 Construction 2: HMAC



S(k,m) = H(k⊕ opad, H(k⊕ipad ||m)) In this way, the mac cannot be extended

3 Overview of symmetric Encryption

3.1 Basic problem

- Given: two parties already share the secret key, the secret key is known only by the two parties
- goal: send a message confidentially

IMPORTANT: Any communication system that aims to guarantee confidentiality must fullfill the basic problem

3.2 One-time pad

- Perfect secrecy
- Disadvantage
    1. May not have secure channel to pass key as long as message

2. key can only use once
3. Does not guarantee integrity
4. Does not meet confussion and diffusion

3.3 Block cipher

3.3.1 Feather

1. Operates on single chunk(block) of plaintext
   - DES 64 bit per block
   - AES 128 bit per block

   Key may be reused in each block
2. Result should look like random permutation
3. Expensive to break(not impossible)
   - No algorithm better than brute-force
   - The cost of breaking may be longer than the value or lifetime of the message

3.4 MAC-then-encrypt, encrypt-then-MAC, Encrypt-and-MAC and Authenticated encryption

3.4.1 Encrypt-then-MAC

Encrypte the plaintext, then compute the MAC on the ciphertecxt, and append it to the ciphertext(In that case, we do not forget to include the initialization vector(IV) and the encryption method identifier into MACed data)
Characters:

1. Provides integrity of cipher text. Assume the MAC shared secret has not been compromosed, we ought to be able to deduce whether a given ciphertext is indeed authentic or has been forged(NOTE: encrypt-then-hash is insecure especially when the cipher scheme is malleable)
2. Plaintext integrity
3. even if the encryption method is malleable, we should not worries because the mac can filter out this invalid ciphertext
4. The MAC does not provide any information on theplaintext since, assuming the output of the cipher appears random, so does the MAC. In other words, nothing about the plaintext

can be found from the MAC

3.4.2 MAC-then-Encrypt

Compute the MAC on the cleartext, append it to the data, and then encrypt the whole(What the TLS does)
reference: https://tools.ietf.org/html/rfc7366

1. Does not provide any integrity on the ciphertext, since we have no way of knowing until we decrypt the message whether it was indeed authentic or spoofed
2. Plaintext integrity, here the mac cannot provide any information of the plaintext either because the plaintext is encrypted
3. If the cipher scheme is malleable, it may be possible to alter the message to appear valid and have a valid MAC. This is a theoretical point. In reality, the MAC serete should provide protection.

3.4.3 Encrypt-and-MAC

Compute the mac on the plaintext, encrypt the plaintext and then append the mac at the end of the ciphertext(What SSH does)

1. No integrity on the ciphertext, becasue the mac is taken against the plaintext. This opens the door to some chosen cipher text attacks on the ciphertext attacks on the cipher
2. The integrity of the plaintext can be verified If the cipher scheme is malleable, the contents of the ciphertext could well be altered, but on decryption, we can find the plaintext is invalid.
3. may reveal information about the plaintext in the mac. This occurs if the plaintext msg are repeated and nouce is not applied(Because the verify of mac is deterministic)

Reference:`https://crypto.stackexchange.com/questions/202/should-we-mac-then-encrypt-or-encrypt-th`

4 Overview of Asymmetric key scheme

4.1 Two problems

4.1.1 Problem 1

- Given: everyone knows Bob's public key and only Bob knows Bob's private key
- Goal:

1. Alice wants to send a message that only Bob can know
2. Bob wants to send a message that only could have written

4.1.2 Problem 2

- Given: no one except Bob knows Bob's public key; Bob knows Bob's private key and public key
- Goal:
  1. Everyone can know Bob's public key
  2. Everyone can verify certain public key is Bob's public key(i.e. integrity of public key)

5 Number theorem for RSA

5.1 Notation

- N denotes a positive integer
- p denotes a prime number
- $\mathbb{Z}_N$ denotes the set 0,1,2,...,N-1

5.2 Definition: ring

A ring is a set R (equipped with two binary operation "+" and "·"(multiplication)satisfying the following three axioms, called ring axioms)

1. Abelian group under addition(Abelian group is also called communicative group, meaning a group in which the result of applying the group operation on the elements does not depend on the order in which they are written)
   - $\forall a, b, c \in R \rightarrow (a + b) + c = a + (b + c)$(associative on +)
   - $\forall a, b \in R \rightarrow a + b = b + a$ (communicative on +)
   - $\exists 0 \in R : \forall a \in R : a + 0 = a$ (identity)
   - $\forall a \in R, \exists -a \in R$ such that $a - a = 0$ (inverse)
2. Monoid under multiplication(where monoid means a set equipped with associative operation and and identity element)
   - $\forall (a \cdot b) \cdot c = a \cdot (b \cdot c)$
   - $\exists 1 \in R : \forall a \in R, a \cdot 1 = a$

3. Multiplicative is distributive with respective to addition
   - $\forall a, b, c \in R, a \cdot (b + c) = a \cdot b + a \cdot c$
   - $\forall a, b, c \in R, (a + b) \cdot c = a \cdot c + b \cdot c$

5.3 Modular arithmetic

- 9+8 = 5 in $\mathbb{Z}_{12}$
- $5 \times 7 = 11$ in $\mathbb{Z}_{12}$
- 5-7 = 10 in $\mathbb{Z}_{12}$

5.4 Greatest Common Divisor(gcd)

1. For integer x,y: c is the greatest common divisor if
   - c|x
   - c|y
   - $\forall z \in \mathbb{Z}, z|x \wedge z|y \rightarrow z \leq c$

   denoted as c=gcd(x,y)

5.4.1 Facts about GCD

$\forall x, y \in \mathbb{Z}, \exists a, b \in \mathbb{Z}$ such that $a \cdot x + b \cdot y = gcd(x, y)$
Proof:

1. Given integers a and b, not both zero, and given d = gcd(a, b), let

   S = {x | x is a positive integer and x = as + bt for some integers s and t}.

2. Note that S is a nonempty set because (1) if $a > 0$ then $1 \cdot a + 0 \cdot b \in$ S, (2) if $a < 0$ then $(-1) \cdot a + 0 \cdot b \in S$, and (3) if a = 0, then by assumption b $\neq$ 0, and hence $0 \cdot a + (-1) \cdot b \in S$ or $0 \cdot a + (1) \cdot b \in S$. Thus, because S is a nonempty subset of pos-itive integers, by the well-ordering principle for the integers there is a least element c in S. By definition of S,

   c = as + bt for some integers s and t.

   We will show that (1)$c \geq d$, and (2) $c \leq d$, and we will therefore be able to conclude that c = d = gcd(a, b).

3. Proof that $c \geq d$:
   [In this part of the proof, we show that d is a divisor of c and thus that d $\leq$ c.]

4. Proof that $c \leq d$:
   [In this part of the proof, we show that c is a divisor of both a and b and therefore that c is less than or equal to the greatest common divisor of a and b, which is d.] (from 2)

5. Therefore we conclude that c = d. It follows that d, the greatest common divisor of a and b, is equal to as + bt.

5.4.2 Euclid algorithm

Can be qpplied to efficiently find the gcd
The Euclidean algorithm provides a very efficient way to compute the greatest common divisor of two integers.
Steps:

1. Let A and B be integers with $A > B \geq 0$

2. To find the greatest common divisor of A and B, first check whether B = 0. If it is, then gcd(A, B) = A. If it isn' t, then $B > 0$ and the quotient- remainder theorem can be used to divide A by B to obtain a quotient q and a remainder r:

   A=Bq+r where $0 \leq r < B$.

3. It can be proved that gcd(A,B)=gcd(B,r), Thus the problem of finding the greatest common divisor of A and B is reduced to the problem of finding the greatest common divisor of B and r.
   What makes this piece of information useful is that B and r are smaller numbers than A and B. To see this, recall that we assumed

   $$A > B \geq 0$$

   Also the r found by the quotient-remainder theorem satisfies

   $$0 \leq r < B$$

   Putting these two inequalities together gives

   $$0 \leq r < B < A.$$

   So the larger number of the pair (B,r) is smaller than the larger number of the pair (A, B).

4. Now just repeat the process, starting again at (2), but use B instead of A and r instead of B. The repetitions are guaranteed to terminate eventually with r = 0 because each new remainder is less than the preceding one and all are nonnegative.

By the way, it is always the case that the number of steps required in the Euclidean algorithm is at most five times the number of digits in the smaller integer. This was proved by the French mathematician Gabriel Lamé (1795–1870).

5.5 Modular inverse

The inverse of x in $\mathbb{Z}_N$ is an element y in $\mathbb{Z}_N$ such that

- The inverse of x in $\mathbb{Z}_N$ is an element y in $\mathbb{Z}_N$ such that

  $x \cdot y = 1$ in $\mathbb{Z}_N$

  y is denoted as x^{-1} in $\mathbb{Z}_N$

5.6 $\mathbb{Z}_N^*$

$\mathbb{Z}_N^*$= {set of invertible elements in $\mathbb{Z}_N$}=$\{x \in \mathbb{Z}_N : gcd(x, N) = 1\}$
proof: See theorem 5.10

5.7 Fermat's theorem

Let p be a prime number, $\forall x \in (\mathbb{Z}_p^*)$, $x^{p-1} = 1$ in $\mathbb{Z}_p$

5.7.1 proof

1. Start by listing the first p-1 positive multiples of a:
   1.1 $a, 2a, 3a, ...(p-1)a$
   1.2 It can be proved that $r \equiv s \mod p \implies ra \equiv sa \mod p$ and $r \neq s \mod p \implies ra \neq sa \mod p$ (a is invertible)
   1.3 Therefore, $\forall x, y \in \{1, 2, ..., p-1\}, x \neq y \implies xa \neq ya \mod p$
   1.4 Therefore $(1a) \cdot (2a) \cdot ... \cdot (p-1)a \equiv 1 \cdot 2 \cdot ... \cdot (p-1) \mod p$
2. Therefore $(1a) \cdot (2a) \cdot ... \cdot (p-1)a \mod p = a^{p-1}(p-1)! \mod p = (p-1)! \mod p$
3. Therefore $(a^{p-1} \mod p = 1 \mod p$

5.8 Euler's theorem: cyclic structure of $\mathbb{Z}_p^*$

Let p be a prime number, $\exists g \in \mathbb{Z}_p^*$ such that $\forall a \in \mathbb{Z}_p^*, \exists k \in \mathbb{Z}$ such that $g^k = a$ in $\mathbb{Z}_p^*$

5.9 Euler's generalization of Fermat(Euler's φ(Phi) function)

For an integer N, define $\varphi(N) = |\mathbb{Z}_N^*|$

i.e. $\varphi(N)$ equals to cardinality of $\mathbb{Z}_N^*$

e.g.

- $\varphi(12) = |1, 5, 7, 11|$
- $\varphi(p) = p - 1$

5.10 Theorem

If p is a prime number, a is a positive integer, $\varphi(p^a) = p^a - p^{a-1}$

5.10.1 Proof

1. Let S denotes set, where if $x \in S \rightarrow$ x$\in \mathbb{Z}_{p^a}$ and $x \notin \mathbb{Z}_{p^a}^*$
2. $\forall x \in S, p|x$ (because p is prime number and gcd(p,x)$\neq$1)

   2.1 therefore, $\exists a \in \mathbb{Z}$ such that x=a·p

   2.1.1 $\because$ x in {0,1,2,...,$p^a - 1$} $\rightarrow a \in A = \{0, 1, 2, ..., p^{a-1} - 1\}$

   2.2 Hence |S|$\leq$ |A|=p^{a-1}

   2.3 $\because \forall a \in A, a \cdot p \in S, \therefore |A| \leq |S|$

   2.4 $\therefore |S| = |A| = p^{a-1}$
3. $\because \mathbb{Z}_p^* = \mathbb{Z}_p \backslash S$
4. $\therefore \varphi(p) = |\mathbb{Z}_p^*| = |\mathbb{Z}_p| - |S| = p^a - p^{a-1}$

5.11 Theorem: φ is multiplicative

if p,q are relative prime number and n=$p \cdot q$, then

$\varphi(n) = \varphi(p) \cdot \varphi(q)$

5.11.1 multiplicative

An arithmetic function f is called multiplicative if f(mn)=f(m)f(n) where m, n are relatively prime

5.11.2 proof

1. Let m,n be relatively prime, $\mathbb{Z}_{mn}$ is denoted as

$$
\begin{array}{rrrcrcr}
\{ & & & & & & \\
0, & 1, & 2, & ..., & c-1, & ..., & m-1, \\
m+0, & m+1, & m+2, & ..., & m+c-1, & ..., & m+m-1, \\
... & & & & & & \\
(r-1)m+0, & (r-1)m+1, & (r-1)m+2, & ..., & (r-1)m+c-1, & ..., & (r-1)m+m-1, \\
... & & & & & & \\
(n-1)m+0, & (n-1)m+1, & (n-1)m+2, & ..., & (n-1)m+c-1, & ..., & (n-1)m+m-1 \\
\} & & & & & &
\end{array}
$$

2. for element on row r and colum c, the value is (r-1)m+(c-1)

   2.1 for any colum c_i, if $gcd(c_i-1,m) \neq 1$, $\forall r \in \{0,1,2,...,n-1\}$:
   $gcd((r-1)m+(c_i-1),m) \neq 1 \rightarrow gcd((r-1)m+(c_i-1),mn) \neq 1$

   2.2 otherwise, for any colum c_i, if $gcd(c_i-1,m) = 1, \forall r \in \{0,1,2,...,n-1\}$:
   $gcd((r-1)m+(c_i-1),m) = 1$ by the Euclidean algorithm;

      2.2.1 It follows from Theorem 4.7 of Rosen that the entries in such a colum c_i form a complete residue system modulo n.

      2.2.2 Thus, exactly φ(n) of them will be relatively prime to n, and thus relatively prime to mn.

3. it's known that there are $\varphi(m)$ such columes by definition of φ, in each one of such column, there are $\varphi(n)$ elements x that satisfy gcd(x,mn)=1

4. therefore, $\varphi(mn) = \varphi(m) \cdot \varphi(n)$

5.11.3 Theorem 4.7 of Rosen

RTBD

5.11.4 Euclidean algorithm

RTBD

5.12 Theorem

$\forall N \in \mathbb{Z}$,define $\varphi(N) = |\mathbb{Z}_N^*|$

$\forall x \in \mathbb{Z}_N^*, x^{\varphi(N)} = 1$ in $\mathbb{Z}_N$

5.12.1 Proof

RTBD

6 Diffie-Hellman Protocol

Given a large prime number b, and a generator g of $\mathbb{Z}_p^*$

$\mathbb{Z}_p^* = \{1, 2, 3, ..., p-1\}$

$\forall a \in \mathbb{Z}_p^*, \exists k \in \mathbb{Z}$, such that $a = g^k$ in $\mathbb{Z}_p$ (By Euler's theorem)

In the Diffie-Hellman, there is a pair x,y and generator g,

let k=$(g^x)y = (g^y)^x$

Hear k can be any value in $\mathbb{Z}_p^*$ or in $\mathbb{Z}_p$(because p is a prime number)

6.1 Whether the distribution of k is well chosen?

RTBD

7 Discrete logarithm(DL) problem

given g^x *mod p*,, it's hard to extract x

- currently, there is no efficient algorithm for doing this
- This is not enough for Deffie-Hellman algorithm to be secure

8 Computational Discrete logarithm(CDL) problem

Given g^x mod p and g^y mod p, it's hard to compute g^{xy} mod p without knowing x and y

9 Decisional Deffie-Hellman Problem

given g^x mod p and g^y mod p, it's hard to tell the difference between g^{xy} mod p and g^r mod p where r is a random number

DDH achieves higher security than CDH and DH
Suppose DDH is a hard problem, Diffie-Hellman protocol is a key estabishment against passive attack

- Estabilished key can be used as symmetric key
- Evesdroper can not tell the difference between the key and a random value

Basic Diffie-Hellman protocol does not provide authentication. In reality, IPSec, forexample, applies signature and anti-DoS cookie

10 Coin flipping

Bob find himself having feeling with Alice, therefore Bob ask Alice to be his girl friend. However, Alice is not sure whether she should accept Bob. Alice's friend Laowang designs a fair game for Alice to decide whether she should accept Bob. In this game, Alice has a coin, which has two side: head and tail.

1. Alice toss drop a coin in a black box with a cap
2. Here we assume that the probability for the coin to be head and tail to be $\frac{1}{2}$(so Laowang is a fair man)
3. Bob guess whether the coin is head up or tail up
4. Alice open the cap(which will not change the state of the coin), and see
    - Bob win if he guess correctly(Alice's boy friend!)
    - Bob loss if he guess wrong(loser)

In this way, to the maximal extent possible, neither Alice nor Bob can cheat the other, and at the same time, each of them learn the outcome of a fair coin toss
Unfortunately, in reality the 1/2 fair coin may not be guaranteed, so an altered solution is that

1. Alice puts a random bit α inside a black envelope and sends it to Bob, Bob cannot open it
2. Bob pick a random bit β
3. Alice open the envelop in front of Bob, and compute $\alpha \oplus \beta$
    - Alice accepts Bob if $\alpha \oplus \beta = 1$
    - Else Alice reject Bob

In computer system, the envelop can be done by encryption, but if we wanna solve it directly using atomic primitive, a solution applying N=pq is as follows

1. Alice pick two large prime number, p and q(where we assume p < q)
    - To put "0" into the envelope, Alice pick p and q such that

$p \equiv 1 \pmod 4$

$q \equiv 3 \pmod 4$

compute N=p*q and give N to Bob

- To put "1" into the envelope, Alice pick p and q such that

  $p \equiv 3 \pmod 4$

  $q \equiv 1 \pmod 4$

  compute N=p*q and give N to Bob

- Poor Bob receives the N, would like to guess whether $p \equiv 3 \pmod 4$ or $p \equiv 1 \pmod 4$, there is no efficient algorithm to factorize N, so Bob unable to get Alice's love with probability more than 2(proof RTBD), so Bob gives a bit β, where 1 represents $p \equiv 3 \pmod 4$ and 0 represents $p \equiv 1 \pmod 4$

- After Bob gives β, Alice gives p and q to Bob

  (a) Bob verify pq=N

  (b) Bob compute p mod 4 to see whether he guessed correctly

11 RSA cryptosystem

11.1 Key generation

1. Generates large prime number p and q, at least 2048 bits each, needing premality test
2. Compute n=$p \cdot q$, notes that $\varphi(n) = \varphi(p)\varphi(q) = (p-1)(q-1)$
3. Choose e, relative prime to $\varphi(n)$
4. compute d such that ed$\equiv$1 mod $\varphi(n)$
5. public key (e,n), private key (d)
6. encryption c=m^e mod n
7. Decryption c^d mod n= m^{ed} mod n

11.1.1 Proof

RTBD

11.2 Facoring problem

Given positive integer n, find $p_1, p_2, ..., p_k$ such that n=$p_1^{e_1} p_2^{e_2} ... p_k^{e_k}$

If factoring problem is easy, then RSA problem is easy.
Currently RSA can be brocken without factoring

11.3 "Textbook" RSA is bad encryption

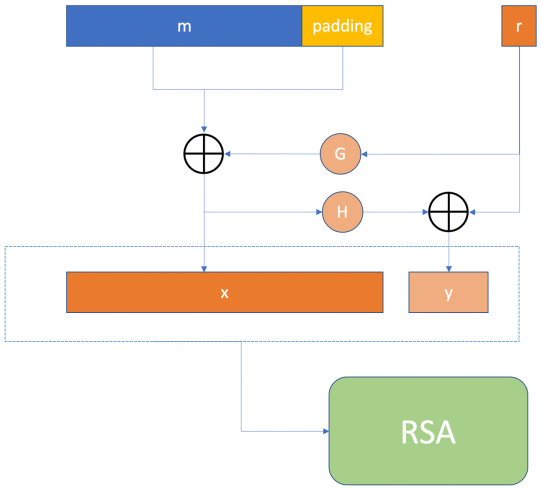
1. Fail against CPA attack without salt
2. If msgs are from small set, can build a table of corresponding cipher text
3. Can tamper with encrypted msg(integrity, malleablity)

   - Take c and submit $c(101/100)^e$ mod n instead(RTBD)

11.4 OAEP(Applied in reality)

11.4.1 Assumption

1. H,G are cryptographic hash function
2. RSA problem is hard
3. r is a random number



$$RSA(M \oplus G(r)|H(M \oplus G(r)))$$

1. padding: fix the length the text

2. r and G: against CPA
3. H and y: protect integrity

11.5 RSA signature

- Given
  1. Everyone knows Bob's real public key
  2. Only Bob knows Bob's private key
  3. A message m is known by Bob and the other party
- Goal
  1. Only man with Bob's private key(Bob) can create signature of m
  2. Man with Bob's public key can verify Bob's signature

11.5.1 Workflow of RSA signature

Public key is (e,n), private key is d

- sign:
  1. $sig(m) = (\underbrace{hash(m)})^d \ mod \ n$
     m is hashed, so unable to derivate derivate information of m from the signature
- verify
  1. given m' and sig, verify whether hash(m')=$sig^e \ mod \ n = hash(m)^{de} \ mod \ n$

11.6 Advantage of RSA Algorithm

1. Confidentiality without pre-shared key
   1.1 There's case where two parties do not have a secure channel to share the secret key
   1.2 RSA can be used to build secure key(note that here is to use RSA to share the secrete symmetric key
   - RSA build secret key: the key is determined by one party only, and encrypted with RSA to share to the other party
   - Deffie-hellman: Both party gives part of information about the key, and the communication process itself makes the key

2. Authentication without shared key

11.7 Limit of RSA

1. Must make the Bob's public key known to every one
2. Slow calculation process: 2-3 orders of magnitude slower. That's why RSA is usually used just for building secret key
3. RSA keys ($\geq$ 2048 bit) are usually longer than symmetric key($\geq$ 128 bit for AES) to achieve the same security
4. Relies on unproven number theoretic assumption(Factoring problem, RSA problem, Discrete logarithm problem, decisional Diffi-Hellman algorith) or generally ($P \neq NP$?)

12 Non-repudiation of public key signature

One standout property of public key signature is that it achieves non-repudiation, which means the party that signed a signature cannot deny that the signature by himself

- MAC does not achieve non-repudiation because in MAC scheme, the party who can verify the mac/tag can also generate the mac/tag
- In public key signature signature, even if everyone with Alice's public key can verify the integrity with the signature, only Alice can sign(so she cannot deny that she signed it)

13 Constructing a secure protocol

- Secure Socket Layer Protocol (SSL)
- Transportation Layer Secure (TLS)

1. Qualitative
   End-to-end secure communication in presence of attacker

   By Kercknoff's principle, attackers own the network, wifi, router, DNS server, the visited website; Can listen to any pkts; Can modify the pkt in transit; Can inject his own pkts into the network(Mealous, Eves)

13.1 History of SSL/TLS

RTBD

13.2 Structure

SSL consists of two parts

1. Handshake protocol uses public-key cryptography to establish shared key between client & server
2. Record protocol: uses the shared key established by handshake to achieve confidentiality, integrity and authentication between client and server

13.3 Workflow of SSH handshake protocol

1. Negociation of versions of protocols between clients and server(Interoperability between different implementation)
2. Authenticate server and client
   - Certificate to learn each other's public key authenticate with public key & signature
   - Often only server is authenticated
3. Use public key to establish shared key