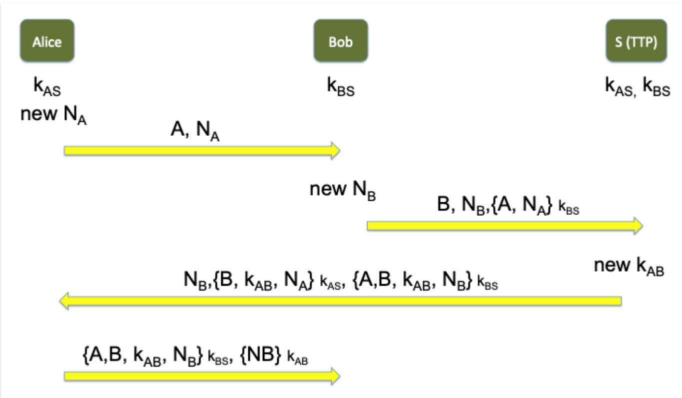
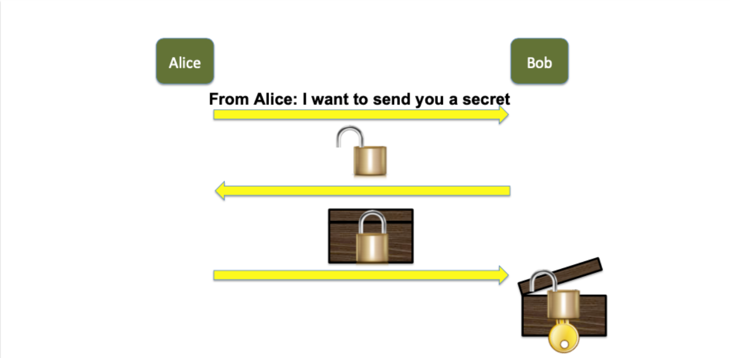
CS Revision Lecture 9, 10

- ## Lecture 9 - Cryptography: asymmetric encryption
  - ### Introduction
    - So far: how two users can protect data using a shared secret key
      - One shared secret key per pair of users that want to communicate
    - Our goal now: how to establish a shared secret key to begin with?
      - Trusted Third Party(TTP)
      - Diffie-Hellman (DH) protocol
      - RSA
      - ElGamal (EG)
  - ### Online Trusted Third Party (TTP)
    - Users $U_1, U_2, U_3, \ldots, U_n, \ldots$
    - **Each user U_i has a shared secret key K_i with the TTP**
    - **Ui and U_j can establish a key $K_{i,j}$ with the help of the TTP**
    - **$\{m\}_k$ denotes the symmetric encryption of m under the key k**
    - Example: using Paulson's variant of the Yahalom protocol



    - Question: can we establish a shared secret key without a TTP?
    - Answer: Yes! Using public key cryptography
  - ### Goal of public-key encryption



- Alice put the secret inside the box
- Alice lock the box using Bob's padlock then send it to Bob
- Bob unlock the padlock using his key and read the secret

- Public-key encryption - Definition
  - Key generation algorithm: $G : \to K \times K$
  - Encryption algorithm $E : K \times M \to C$
  - Decryption algorithm $D : K \times C \to M$
  - st. $\forall (sk, pk) \in G$, and $\forall m \in M$, $D(sk, E(pk, m)) = m$

  - **The decryption key sk_{Bob} is secret (only known to Bob).** The encryption key pk_{Bob} is known to everyone. And $sk_{Bob} \neq pk_{Bob}$

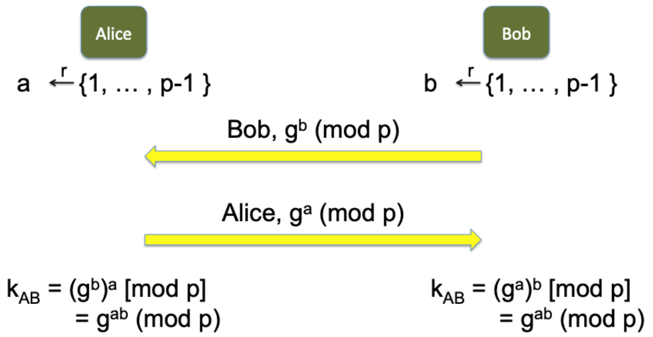
- Primes
  - Definition
    - p $\in \mathbb{N}$ is a **prime** if its only divisors are 1 and *p*
    - Ex: 2, 3, 5, 7, 11, 13, 17, 19, 23, 29
  - Theorem

- Every $n \in \mathbb{N}$ has a **unique factorization** as a product of prime numbers (which are called its factors)
- Ex: 23244 = 2 x 2 x 3 x 13 x 149

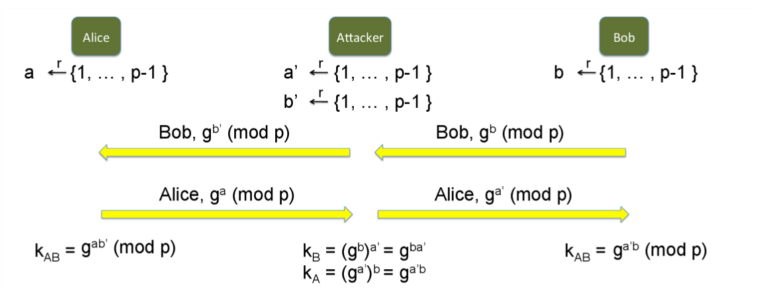
- Relative primes
  - **Definition**
    - a and b in $\mathbb{Z}$ are **relative primes** if they have no common factors
  - Euler function
    - The Euler function $\phi(n)$ is the **number of elements** that are relative primes with n:
      - $\phi(n) = |\{m|0 < m < n$ and gcd(m, n) = 1$\}|$
      - For p prime: $\phi(p) = p - 1$
      - **For p and q primes: $\phi(p \cdot q) = (p-1)(q-1)$**
- $\mathbb{Z}_n$
  - Let $n \in \mathbb{N}$. We define $\mathbb{Z}_n = \{0, ..., n-1\}$
    - $\forall a \in \mathbb{Z}, \forall b \in \mathbb{Z}_n, a \equiv b(mod\ n) \iff \exists k \in \mathbb{N}.\ a = b + k \cdot n$
  - Modular inversion:
    - the inverse of $x \in \mathbb{Z}_n$ is $y \in \mathbb{Z}_n$ s.t. $x \cdot y \equiv 1(mod\ n)$. We denote x^{-1} the inverse of x mod n
      - Example:
        - $7^{-1}\ in\ \mathbb{Z}_{12}\ :\ 7$ 7 * 7 = 49 mod 12 = 1
        - $4^{-1}\ in\ \mathbb{Z}_{12}$: 4 has no inverse in $\mathbb{Z}_{12}$
  - Theorem
    - Let $n \in \mathbb{N}$. Let $x \in \mathbb{Z}_n$. x has a inverse in $\mathbb{Z}_n$, iff gcd (x, n) = 1
- $\mathbb{Z}_n^*$
  - Let $n \in \mathbb{N}$. We define $\mathbb{Z}_n^* = \{x \in \mathbb{Z}_n | gcd(x, n) = 1\}$
    - Example: $\mathbb{Z}_{12}^* = \{1, 5, 7, 11\}$
  - Note that $|\mathbb{Z}_n^*| = \phi(n)$
    - Number of prime numbers

- Theorem (Euler)
  - $\forall n \in \mathbb{N}, \forall x \in \mathbb{Z}_n^*$, if gcd(x,n) = 1 then $x^{\phi(n)} \equiv 1(mod\ n)$
    - Ex:
      - $11^{12}\ mod\ 12 = 1$
      - $7^{12}\ mod\ 12 = 1$
      - $5^{12}\ mod\ 12 = 1$
      - $1^{12}\ mod\ 12 = 1$
  - $\forall p$ prime, $\mathbb{Z}_p^*$ is a cyclic group, i.e.
    - $\exists g \in \mathbb{Z}_p^*, \{1, g, g^2, g^3, ..., g^{p-2}\} = \mathbb{Z}_p^*$
    - Ex:
      - p = 7, $\mathbb{Z}_7^* = \{1, 2, 3, 4, 5, 6\}$
      - g = 3, s.t. $\mathbb{Z}_7^* = \{1, 3\ mod\ 7, 3^2\ mod\ 7, 3^3\ mod\ 7, 3^4\ mod\ 7, 3^5\ mod\ 7\} = 1, 3, 2, 6, 4, 5$
- Intractable problem
  - Factoring:
    - input: n $\in \mathbb{N}$
    - output: $p_1, ..., p_m$ primes s.t. $n = p_1, ..., p_m$
  - RSAP
    - input
      - n st. $n = p \cdot q$ with 2 $\leq p, q$ primes
      - e st. $gcd(e, \phi(n)) = 1$
      - m^e mod n
    - output:
      - m
  - Discrete Log:
    - Input: prime p, generator g of $\mathbb{Z}_p^*, y \in \mathbb{Z}_p^*$
    - Output: x such that $y = g^x(mod\ p)$
  - DHP (Diffie-Hellman problem)

        - Input: prime p, generator g of $\mathbb{Z}_p^*$, $g^a(mod\ p)$, $g^b(mod\ p)$
        - Output: $g^{ab}(mod\ p)$
- The Diffie-Hellman (DH) Protocol
    - **Assumption: the DHP is hard in $\mathbb{Z}_p^*$**
    - Fix a very large prime p, and $g \in \{1, ..., p-1\}$



    - It is hard to know $g^{ab}(mod\ p)$ because of DHP
    - Man-in-the-middle attack



        - Attacker create number a' and b'.
        - Send them to Alice and Bob. Create keys that attacker knows
- RSA trapdoor permutation
    - $G_{RSA}() = (pk, sk)$
        - Where $pk = (N, e)$ and $sk = (N, d)$
        - $N = p \cdot q$ with p, q **random primes**
        - $e, d \in \mathbb{Z}$ st. $e \cdot d = 1 + k \cdot \phi(N) \equiv 1(mod\ \phi(N))$
    - $M = C = \mathbb{Z}_N$

- $RSA(pk, x) = x^e (mod\ N)$
- $RSA^{-1}(sk, x) = x^d (mod\ N)$
- Consistency:
    - $\forall (pk, sk) = G_{RSA}(), \forall x, RSA^{-1}(sk, RSA(pk, x)) = x$
    - Proof:
        - Let $pk = (N, e)$, $sk = (N, d)$ and $x \in \mathbb{Z}_N$. Easy case where x and N are relatively prime

$$\begin{aligned} RSA^{-1}(sk, RSA(pk, x)) &= (x^e)^d \pmod N \\ &= x^{e \cdot d} \pmod N \\ &= x^{1+k\phi(N)} \pmod N \\ &= x \cdot x^{k\phi(N)} \pmod N \\ &= x \cdot (x^{\phi(N)})^k \pmod N \\ &\overset{\text{Euler}}{=} x \pmod N \end{aligned}$$

- How Does it work
    - choose two large prime numbers $p\ and\ q$
    - $N = p \cdot q$
    - $\phi(N) = (p-1) \cdot (q-1)$ Euler function
    - Choose e **(encryption key)**
        - $1 < e < \phi(N)$
        - e coprime with $N, \phi(N)$
        - e is public
    - Choose d **(decryption key)**
        - $e \cdot d(mod\ \phi(N)) = 1$
        - d is private

- How **NOT** to use RSA
    - (G_{RSA}, RSA, RSA^{-1}) is called raw RSA. Do not use raw RSA directly as an asymmetric cipher
        - **RSA is deterministic $\implies$ not secure against chosen plaintext attacks**

          No randomness at all

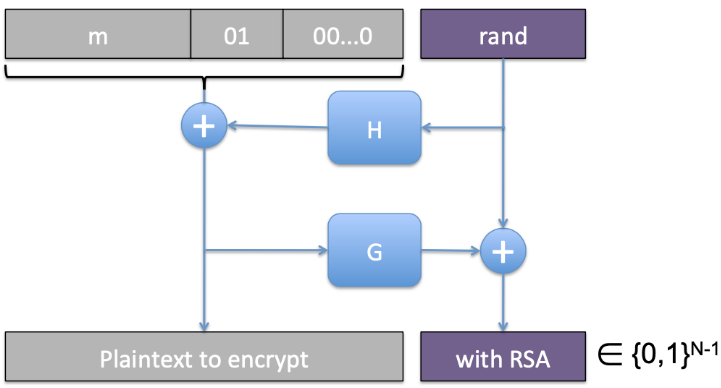
- ISO Standard
  - Goal:
    - Build a CPA secure asymmetric cipher using (G_{RSA}, RSA, RSA^{-1})
  - Let (E_s, D_s) be a symmetric encryption scheme over (M, C, K)
  - Let $H : Z_N^* \to K$

    Hash function produce the Key
  - **Build $(G_{RSA}, E_{RSA}, D_{RSA})$ as follows**
    - G_{RSA} as described above
    - $E_{RSA}(pk, m)$:
      - pick random $x \in \mathbb{Z}_N^*$
      - $y \leftarrow RSA(pk, x)(= x^e \ mod \ N)$

        Encrypt x produce y
      - $k \leftarrow H(x)$

        produce key by putting x into the hash function
      - $E_{RSA}(pk, m) = y||E_s(k, m)$
    - $D_{RSA}(sk, y||c) = D_s(H(RSA^{-1}(sk, y)), c)$

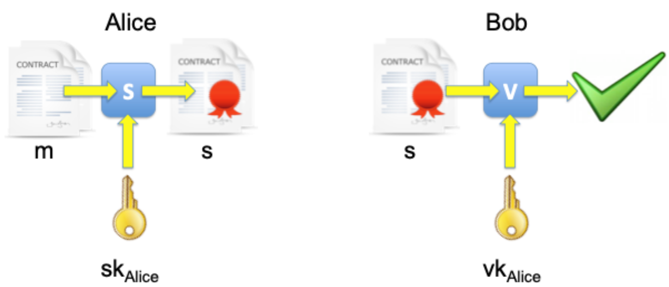
      First recover the x, then decrypt the ciphertext
- PKCS1 v2.0: RSA-OAEP
  - Goal: build a CCA (chosen ciphertext attacks) secure asymmetric cipher using (G_{RSA}, RSA, RSA^{-1})



- ElGamal (EG)
  - Fix prime p, and generator $g \in \mathbb{Z}_p^*$
  - $M = \{0, ..., p-1\}$ and $C = M \times M$
  - $G_{EG}() = (pk, sk)$
    - $pk = g^d (mod\ p)$
    - $sk = d$ and $d \xleftarrow{r} \{1, ..., p-2\}$
  - $E_{EG}(pk, x) = (g^r (mod\ p), m \cdot (g^d)^r (mod\ p)$
    - $r \xleftarrow{r} \mathbb{Z}$
  - $D_{EG}(sk, x) = e^{-d} \cdot c(mod\ p)$
    - $x = (e, c)$
  - Consistency:
    - $\forall (pk, sk) = G_{EG}(), \forall x, D_{EG}(sk, E_{EG}(pk, x)) = x$
    - Proof:
      - Let $pk = g^d (mod\ p)$ and $sk = d$

$$\begin{aligned} D_{EG}(sk, E_{EG}(pk, x)) &= (g^r)^{-d} \cdot m \cdot (g^d)^r \pmod{p} \\ &= m \pmod{p} \end{aligned}$$

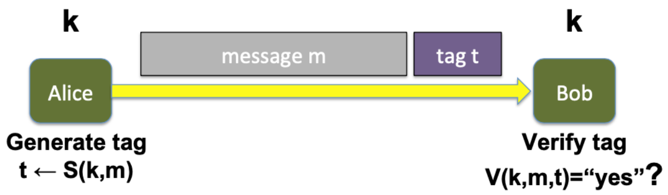
- **Lecture 10 - Cryptography: digital signatures**
  - Goal
    - Data integrity and origin authenticity in the public-key setting



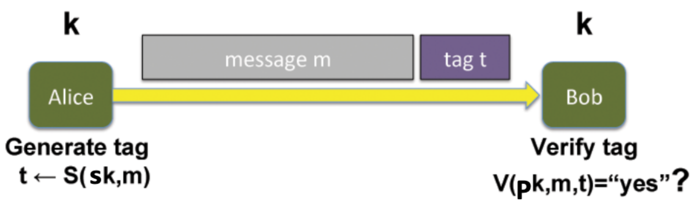
    - Key generation algorithm: $G :\to K \times K$
    - signing algorithm $S : K \times M \to S$

- Verification algorithm $V : K \times M \times S \rightarrow \{\top, \bot\}$
- s.t. $\forall(sk, vk) \in G$, and $\forall m \in M$, $V(vk, m, S(sk, m)) = \top$

- Advantages of digital signatures over MACs
  - MACs



    - **are not publicly verifiable (and so not transferable)**
      - No one else, except Bob, can verify t.
    - **do not provide non-repudiation**
      - t is not bound to Alice's identity only. Alice could later claim she didn't compute t herself. It could very well have been Bob since he also knows the key k
  - Digital signatures



    - are **publicly verifiable** -anyone can verify a signature
    - are **transferable** - due to public verifiability
    - provide **non-repudiation** - if Alice signs a document with her secret key, she cannot deny it later

- Security
  - A good digital signature schemes should satisfy existential unforgeabitliy.
  - What is Existential unforgeability
    - Given $(m_1, S(sk, m_1)), \ldots, (m_n, S(sk, m_n))$ (where $m_1, \ldots, m_n$ chosen by the adversary)

- It should be hard to compute a valid pair $(m, S(sk, m))$ without knowing sk for any $m \notin \{m_1, ..., m_n\}$

- Textbook RSA signatures
  - $G_{RSA}() = (pk, sk)$
    - Where $pk = (N, e)$ and $sk = (N, d)$
    - $N = p \cdot q$ with p, q **random primes**
    - $e, d \in \mathbb{Z}$ st. $e \cdot d = 1 + k \cdot \phi(N) \equiv 1(mod\ \phi(N))$
  - $M = C = \mathbb{Z}_N$
  - Signing: $S_{RSA}(sk, x) = (x, x^d(mod\ N))$
  - Verifying $V_{RSA}(pk, m, x) =$
    - $\top$ if $m = x^e(mod\ N)$
    - $\bot$ otherwise
  - s.t. $\forall(pk, sk) = G_{RSA}(), \forall x, V_{RSA}(pk, x, S_{RSA}(sk, x)) = \top$
  - Proof: exactly as proof of consistency of RSA encryption/decryption

- Problems with "Textbook RSA signatures"
  - **Textbook RSA signatures are not secure**
    - The “textbook RSA signature” scheme does not provide **existential unforgeabitlity**
  - Suppose Eve has two valid signatures $\sigma_1 = M_1^d\ mod\ n$ and $\sigma_2 = M_2^d\ mod\ n$ from Bob, on messages M_1 and M_2.
  - Then Eve can exploit the homomorphic properties of RSA and produce a new signature
    - $\sigma = \sigma_1 \cdot \sigma_2\ mod\ n = M_1^d \cdot\ M_2^d\ mod\ n = (M_1 \cdot M_2)^d\ mod\ n$

- which is a valid signature from Bob on message $M_1 \cdot M_2$

- How to use RSA for signatures
  - **Solution**
    - Before computing the RSA function, apply a hash function H
  - Signing: $S_{RSA}(sk, x) = (x, H(x)^d \pmod N)$
  - Verifying: $V_{RSA}(pk, m, x) =$
    - $\top$ if $H(M) = x^e \pmod N$
    - $\perp$ otherwise