# Tutorial 2 Crypto: Public Key Encryption

Teodora Baluta

March 16, 2018

# What is Public Key Crypto?

- Alice and Bob do not have a pre-shared key, $K$
- Instead, each have a pair of public and private key $(K_{pub}, K_{priv})$
- If Bob has $(K_{pubB}, K_{privB})$, anyone can send a message $m$ to Bob by encrypting with his public key $K_{pubB}$
  - $E(K_{pubB}, m) = c$
- Only Bob can decrypt using his private key $K_{privB}$
  - $D(K_{privB}, c) = m$ (remember *consistency equation*)
- Also known as *asymmetric encryption*

# Some history

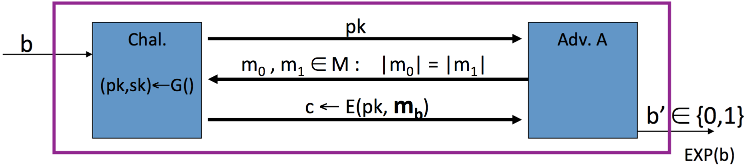
- Concept was introduced in 1976 by Diffie and Hellman in New Directions in Cryptography
- But was proposed in 1973 by Clifford Cocks in a classified paper made public in 1997
- Diffie and Hellman won the 2015 Turing Award

# Examples

- RSA - based on hardness of factoring of large number
- ElGamal - based on hardness of computing the discrete logarithm
- Use case: Email Security & OpenPGP
    - anyone who has your public key can send you encrypted emails

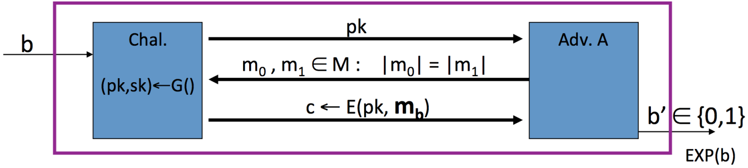
# Is public key crypto secure from eavesdropping?

- Adversary gets to choose random PT $(m_0, m_1)$ to encrypt and receives CT $c$



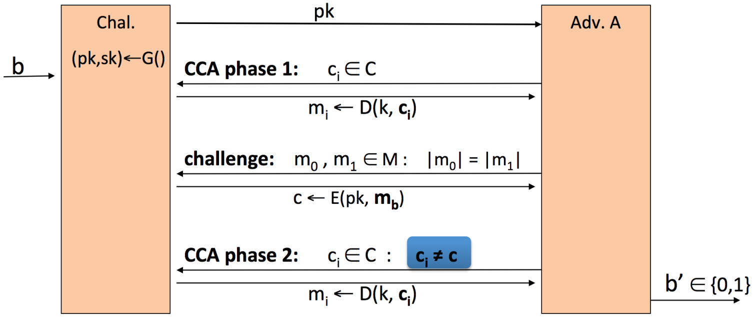

- Adversary gets to eavesdrop and receives one CT but has to guess whether he receives encryption of $m_0$ or encryption of $m_1$

# Semantic Security: eavesdropping

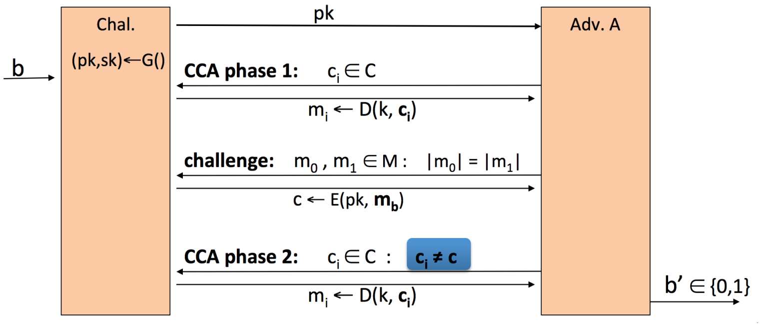



The public key encryption has semantic security if attacker cannot distinguish EXP(0) from EXP(1), i.e. if he got the encryption of $m_0$ or $m_1$

# Chosen-ciphertext Attack (CCA)

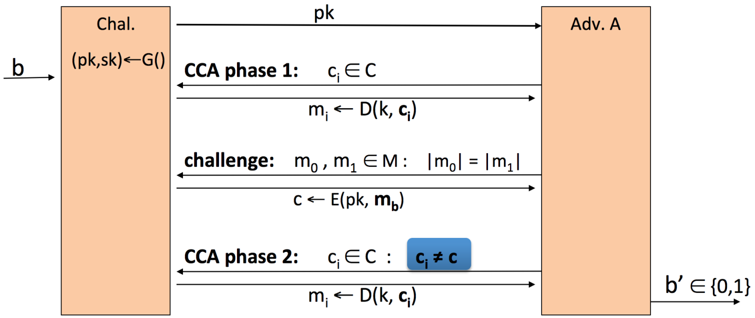


# Chosen-ciphertext Attack (CCA)



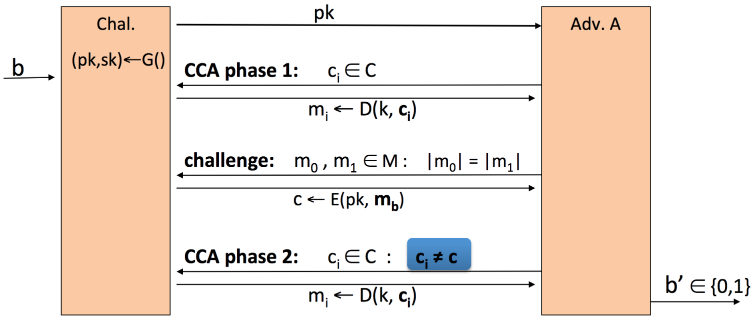

- $b$ is either 0 or 1 and corresponds to one of the experiments EXP(0) or EXP(1)
- challenger gives public key to adversary
- CCA phase 1: adversary sends out ciphertext $c_1$ and gets decryption $m_1$, then $c_2$ and gets $m_2$ ... (he can submit as many as he wants)

# Chosen-ciphertext Attack (CCA)



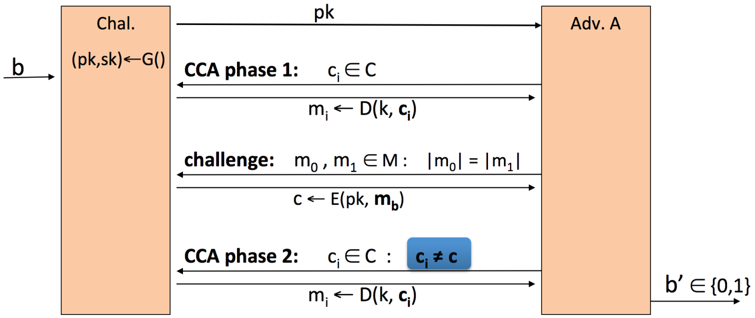

- challenge phase: as normal, adversary submits 2 messages of equal length and challenger sends $c$ which can be either $m_0$ or $m_1$ depeding on $b$

# Chosen-ciphertext Attack (CCA)




- CCA phase 2: adversary can continue to send decryption queries but only for ciphertexts different than the challenge one
- attacker has to say which plaintext corresponds to the challenge ciphertext $c$

# Chosen-ciphertext Attack (CCA)




- Public key encryption scheme is CCA secure if attacker's guess at the game is as good as randomly guessing

# Recap: modulo arithmetic

- let $N = pq$ where p and q are prime numbers
- $Z_N = \{0, 1, ...N - 1\}$ (all operations modulo N)
- $(Z_N)^* = \{\text{invertible elements in } Z_N\}$
- invertible element $x \in Z_N$ iff $gcd(x, N) = 1$
- totient function
  $\phi(N) = (p - 1)(q - 1) = pq - p - q + 1 = N - (p + q) + 1$
- $|(Z_N)^*| = \phi(N)$
- any $x \in (Z_N)^*$ then $x^{\phi(N)} = 1(\text{mod } N)$

# Textbook RSA

- Key generation:
  - select 2 large prime numbers $p$ and $q \approx 1024$ bits unknown to attacker
  - compute $N = pq$ and totient $\phi(N) = (p-1)(q-1)$
  - choose $e, d$ such that $e \cdot d = 1(\mathrm{mod}\phi(N))$
  - Public key is $(e, N)$, private key is $(d, N)$
- Encryption
  - $E(m, (e, N)) = m^e$ in $Z_N$
- Decryption
  - $D(c, (d, N)) = c^d$ in $Z_N$
  - $D(c, (d, N)) = c^d = (m^e)^d = m^{ed} = m^{k \cdot \phi(N)+1} = (m^{\phi(N)})^k \cdot m = m$ in $Z_N$

What we just showed is "textbook RSA"

# Textbook RSA

- textbook RSA is deterministic – not semantically secure
  - if the same message is encrypted twice you know it
  - say you do traffic analysis and see E('yes'), E('yes') and then E('no'), can tell the third message is different
- RSA is a *trapdoor* one-way permutation
  - easy to compute $y = f(x)$ given x and the public key
  - but it is difficult to compute $f^{-1}(y)$
  - however with the private key (the trapdoor) we can easily compute $f^{-1}(y)$

# Textbook RSA

- $e^{th}$ root attack:
  - for short messages $m$ and low $e$ (e.g. e=3), it may happen that $m^e < N$
  - $c = E(m, (e, N) = m^e (\mathrm{mod} N)$ so that the ciphertext is actually $c = m^e$ and then it is possible to implement the decryption function as $e$-th root extraction.

# RSA

In practice, RSA uses more complex constructions (standardized: PKCS) that introduce some form of randomized padding that have better security guarantees under CCA adversary model.

# Diffie-Hellman Key Exchange

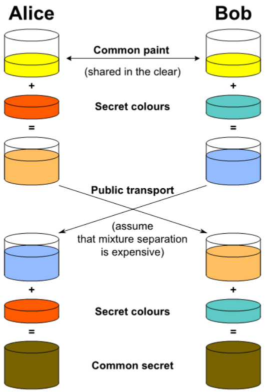



Figure: Diffie Hellman Exchange, Wikipedia

# Diffie-Hellman Key Exchange

- Let G be a finite cyclic group (for example $(Z_p)*$) of order n
- Fix generator $g$ in G: $G = \{1, g, g^2, ...g^{n-1}\}$

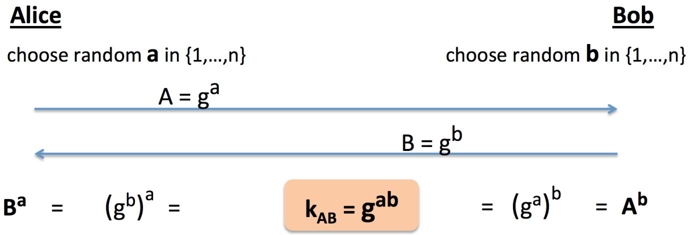



Figure: Diffie Hellman Exchange, Dan Boneh's course

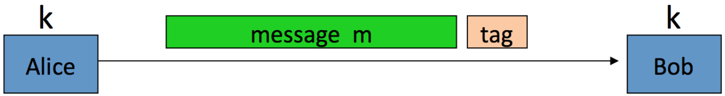
- g, $g^a$, $g^b$ are public
- no efficient way to compute $g^{ab}$ given the above in group G

# ElGamal

- large prime $p$ and a primitive root (or generator) $g$ of group QR($Z_p^*$)
- Key Generation Bob sends to Alice
  - Bob selects a private key a, and generates his public key $\beta = g^a (modp)$
  - publishes (g, p, $\beta$) publicly
- Encryption Alice
  - Alice chooses a random secret k and computes $r = g^k (modp)$
  - Alice computes $t = \beta^k \cdot m(modp)$
  - Alice sends the ciphertext (r, t) to Bob
- Decryption Bob using his private key a
  - D(c, pub) $= t \cdot r^{-a} (modp) =$ m
- ElGamal is not CCA secure
- more complex variants of ElGamal exist

# Sneak peak into Integrity

## Message integrity: MACs




**Generate tag:**
**tag ← S(k, m)**

**Verify tag:**
**V(k, m, tag) $\stackrel{?}{=}$ `yes'**

Def: **MAC** I = (S,V) defined over (K,M,T) is a pair of algs:

- S(k,m) outputs t in T
- V(k,m,t) outputs `yes' or `no'

Backup Slides

# Other types of attacks

- known-plaintext attack: attacker knows random ciphertext-plaintext pairs but doesn't get to choose
- chosen-plaintext attack: attacker chooses ciphertexts and can get the corresponding plaintexts
- known-ciphertext attack: attacker is given some ciphertext, but does not know what the plaintext corresponding to this ciphertext is
- chosen-ciphertext attack: the attacker can choose any ciphertext and obtain the corresponding plaintext