Lecture 19: Public-key Cryptography (Diffie-Hellman Key Exchange & ElGamal Encryption)

Recall

- In private-key cryptography the secret-key sk is always established ahead of time
- The secrecy of the private-key cryptography relies on the fact that the adversary does not have access to the secret key sk
- For example, consider a private-key encryption scheme
  1. The Alice and Bob generate $\mathsf{sk} \xleftarrow{\$} \mathsf{Gen}()$ ahead of time
  2. Later, when Alice wants to encrypt and send a message to Bob, she computes the cipher-text $c = \mathsf{Enc}_{\mathsf{sk}}(m)$
  3. The eavesdropping adversary see c but gains no additional information about the message m
  4. Bob can decrypt the message $\widetilde{m} = \mathsf{Dec}_{\mathsf{sk}}(c)$
  5. Note that the knowledge of sk distinguishes Bob from the eavesdropping adversary

Perspective

- If $|\mathsf{sk}| \geqslant |m|$, then we can construct private-key encryption schemes (like, one-time pad) that is secure even against adversaries with unbounded computational power
- If $|\mathsf{sk}| = O(|m|^{\varepsilon})$, where $\varepsilon \in (0,1)$ is a constant, then we can construction private-key encryption schemes using pseudorandom generators (PRGs)
- What if, $|\mathsf{sk}| = 0$? That is, what if Alice and Bob never met? How is "Bob" any different from an "adversary"?

In this Lecture

- We shall introduce the Decisional Diffie-Hellmann (DDH) Assumption and the Diffie-Hellman key-exchange protocol,
- We shall introduce the El Gamal (public-key) Encryption Scheme, and
- Finally, abstract out the principal design principles learned.

Decisional Diffie-Hellman (DDH) Computational Hardness Assumption I

- Let $(G, \circ)$ be a group of size N that is generated by g. We represent it as $(G, \circ) = \langle g \rangle$.
  - We shall represent $g^0 = e$, the identity of the group $(G, \circ)$
  - We shall use the short-hand to represent $g^i = \overbrace{g \circ g \circ \cdots \circ g}^{i\text{-times}}$
  - Then, we have the set $G = \{g^0, g^1, g^2, \dots, g^{N-1}\}$
  - We have already seen how to compute g^a efficiently, for $a \in \{0, 1, \dots, N-1\}$ using repeated squaring
  - We can easily compute the $\mathrm{inv}(g^a)$ (Think)
- Note that we are not providing the entire set G written down as a set. This has N entries and is too long (for intuition, think of N as 1024-bit number, so N is roughly 2^{1024}). We only provide a succinct way to generate the group G by providing the generator g. Given i, we can efficiently generate the element $g^i \in G$

Decisional Diffie-Hellman (DDH) Computational Hardness Assumption II

Definition (Decisional Diffie-Hellman Assumption)

There exists groups $(G, \circ) = \langle g \rangle$ such that no computationally-bounded adversary can efficiently distinguish the following two distributions

- The distribution of $(A = g^a, B = g^b, C = g^{ab})$, where $a, b \xleftarrow{\$} \{0, 1, \ldots, N-1\}$, and
- The distribution of $(A = g^a, B = g^b, R = g^r)$, where $a, b, r \xleftarrow{\$} \{0, 1, \ldots, N-1\}$

Decisional Diffie-Hellman (DDH) Computational Hardness Assumption III

Remarks:

- Note that DDH Assumption is a "belief" and not a "fact." If it is proven that such groups exist where DDH assumption holds, then this proof will also imply that P $\neq$ NP
- We emphasize that the DDH assumption need not hold for an arbitrary group. There are specially constructed groups where DDH assumption is believed to hold
- For a fixed value of $A = g^a$ and $B = g^b$, note that there is a unique value of $C = g^{ab}$
- The definition, intuitively, states that "Even given $A = g^a$ and $B = g^b$, the adversary cannot (efficiently) distinguish $C = g^{ab}$ from a random $R = g^r$." Alternatively, "even given $A = g^a$ and $B = g^b$, the element $C = g^{ab}$ looks random to a computationally bounded adversary."

Decisional Diffie-Hellman (DDH) Computational Hardness Assumption IV

- Note that it is implicit in the DDH assumption that given $A = g^a$ and g, it is computationally inefficient to compute $a = \log_g A$, i.e., computing the discrete logarithm is hard in the group (Think: Will DDH hold in a group if computing the discrete logarithm is easy?)
- Note that if $a = 0$ (i.e., $A = e$) then it is clear that $C = g^{ab} = e$ as well. Then the adversary can distinguish between g^{ab} and g^c (random c). However, it is unlikely that $a = 0$ (or, $b = 0$) will be chosen. It is possible that there are particular values of a and b when an adversary can distinguish $C = g^{ab}$ from $R = g^r$, but the DDH assumption says that those bad values of a and b are rare, and, consequently, unlikely to be chosen. Thus, it is extremely crucial that a, b are picked at random from the set $\{0, 1, \ldots, N-1\}$

- We shall present an example group where DDH Assumption is clearly false
- Let p be a prime and consider the group $(\mathbb{Z}_p^*, \times)$, where $\times$ is integer multiplication $\mod p$
- Let g be a generator for this group. That is, we have $\{g^0, g, \dots, g^{p-2}\}$ is identical to the set $\{1, 2, \dots, p-1\}$
- Given $X = g^x$, for $x \in \{0, 1, \dots, p-2\}$, we can efficiently determine whether x is even or not! (Note: We shall not compute x. We shall only determine whether x is even or not.)
    - Here is the algorithm. The case of $p = 2$ is easy. Suppose $p > 2$.
    - Note that if $x = 2k$ (that is, x is even), then $X^{(p-1)/2} = \left(g^{2k}\right)^{(p-1)/2} = \left(g^{p-1}\right)^k = 1^k = 1$.

- Note that if $x = 2k+1$ (that is, x is odd), then $X^{(p-1)/2} = \left(g^{2k+1}\right)^{(p-1)/2} = \left(g^{p-1}\right)^k g^{(p-1)/2} = 1^k g^{(p-1)/2} = g^{(p-1)/2}$. Note that g is a generator of $\mathbb{Z}_p^*$, so $g^{(p-1)/2} \neq 1$ (because the smallest power $t > 0$ for which $g^t = 1$ is $t = p-1$). So, we conclude $X^{(p-1)/2} \neq 1$.
- So, given $X \in \mathbb{Z}_p^*$, we can (efficiently compute and) check $X^{(p-1)/2} = 1$ or not. This test identifies whether x is even or not, where $X = g^x$

- For brevity, we shall say that X is an even power, if $X = g^x$ and x is even. Similarly, we shall say that X is an odd power, if $X = g^x$ and x is odd.
- So, given A and B we can determine if A or B is an even power. If A or B is an even power then C is an even power as well! However, the element R shall be an even power only with probability $1/2$.

Example: Group where DDH Assumption does NOT hold III

- We can use this observation to efficiently distinguish samples from the distribution (A, B, C) from (A, B, R). Suppose we are given elements (α, β, γ). We perform the following test

$$\big(\text{Is } (\alpha \text{ or } \beta) \text{ an even power}\big) \text{ and Is } \gamma \text{ an even power}$$

- Suppose $(\alpha, \beta, \gamma) \sim (g^a, g^b, g^{ab})$, where $a, b \xleftarrow{\$} \{0, 1, \dots, N-1\}$. Note that the probability that α or β is an even power is $3/4$. Conditioned on α or β being an even power, the probability that γ is an even power is 1. So, the probability that this test returns true is $(3/4) \cdot 1 = 3/4$.
- Suppose $(\alpha, \beta, \gamma) \sim (g^a, g^b, g^r)$, where $a, b, r \xleftarrow{\$} \{0, 1, \dots, N-1\}$. Note that the probability that α or β is an even power is $3/4$. Conditioned on α or β being an even power, the probability that γ is an even power is $1/2$. So, the probability that this test returns true is $(3/4) \cdot (1/2) = 3/8$.

Example: Group where DDH Assumption does NOT hold IV

- So, this test distinguishes the distribution (A, B, C) from (A, B, R).

Example: Group where DDH is believed to hold

- Let p and q be primes such that $p = 2q + 1$
- Let g be a generator of the group $(\mathbb{Z}_p^*, \times)$, where $\times$ is integer multiplication mod p
- Let G' be the set of all even powers in G. That is, we have $G' = \{g^0, g^2, \ldots, g^{p-3}\}$.
- Now, for large primes p the DDH assumption is believed to hold in the group $(G', \times)$, where $\times$ is integer multiplication mod p

DDH Key-Agreement Protocol I

Alice		Bob
$a \xleftarrow{\$} \{0, 1, \ldots, N-1\}$		$b \xleftarrow{\$} \{0, 1, \ldots, N-1\}$
$A = g^a$		$B = g^b$
	$\xleftarrow{\quad B \quad}$	
	$\xrightarrow{\quad A \quad}$	
Compute sk $= B^a$		Compute sk $= A^b$

DDH Key-Agreement Protocol II

- Note that both parties can computed the key g^{ab}
- An adversary sees $A = g^a$ and $B = g^b$. From this adversary's perspective, the key g^{ab} is indistinguishable from the random element g^r. So, the key $\mathsf{sk} = g^{ab}$ is hidden from the adversary

DDH Key-Agreement Protocol III

Remarks.

- Why is this algorithm efficient? Alice can compute A from the generator g and a using the "repeated squaring technique." Similarly, Alice can also compute the key $\mathsf{sk} = B^a$ by repeated squaring technique.
- What advantage does the parties have over the adversary? Alice knows a, therefore she can compute A and B^a efficiently. Bob knows b, therefore he can compute B and A^b efficiently. Adversary, however, only sees A and B, and DDH states that it is computationally infeasible to distinguish g^{ab} from a random group element g^r. Note that if the adversary can compute the discrete log $\log_g A$, then she can easily compute $B^{(\log_g A)}$, the key.

How to use the Secret Key

- At the end of the Diffie-Hellman key-exchange protocol, Alice and Bob has established a secret key sk that is hidden from the adversary
- Note that Alice and Bob did not have to meet earlier to establish this secret key (contrast this with the private-key encryption scenario, where Alice and Bob have to meet first to establish a secret-key sk)
- Now, we can use the key sk generated by the Diffie-Hellman key-exchange protocol and run any private-key cryptographic primitive using the secret key sk
    - The benefit is that Alice and Bob did not have to meet earlier
    - The downside is that the scheme is secure only against computationally bounded adversaries

Summary of this Scheme. Run the one-time pad private-key encryption over the group $(G, \circ)$ using the key generate by the Diffie-Hellman key-exchange protocol.

ElGamal Public-key Encryption II

Recall the Diffie-Hellman key-exchange protocol.

Alice		Bob
$a \xleftarrow{\$} \{0, 1, \ldots, N-1\}$		$b \xleftarrow{\$} \{0, 1, \ldots, N-1\}$
$A = g^a$		$B = g^b$
	$\xleftarrow{\quad B \quad}$	
	$\xrightarrow{\quad A \quad}$	
Compute sk $= B^a$		Compute sk $= A^b$

ElGamal Public-key Encryption III

- To encrypt a message $m \in G$, Alice encrypts as follows $c = m \circ \mathsf{sk} = m \circ g^{ab}$
- To decrypt a cipher-text $c \in G$, Bob decrypts as follows $\widetilde{m} = c \circ \mathrm{inv}(\mathsf{sk}) = c \circ g^{-ab}$

ElGamal Public-key Encryption IV

We summarize this protocol (ElGamal Encryption) below.

Alice		Bob
		$b \xleftarrow{\$} \{0, 1, \ldots, N-1\}$
		$B = g^b$
	$\xleftarrow{\quad B \quad}$	
$a \xleftarrow{\$} \{0, 1, \ldots, N-1\}$		
$A = g^a$		
Compute $c = m \circ B^a$		
	$\xrightarrow{\quad (A, c) \quad}$	

ElGamal Public-key Encryption V

- The element B sent by Bob is Bob's public-key. It is announced to the world by Bob only once.
- Whoever wants to send an encrypted message to Bob, uses Bob's public-key B
- The pair of elements (A, c) sent by Alice is the cipher-text
- Bob can easily decrypt by computing $\widetilde{m} = c \circ \mathrm{inv}(A^b)$
- The algorithm followed by Alice is her encryption algorithm. To encrypt a new message m', Alice will choose a fresh random a' and compute $A' = g^{a'}$ and $c' = m' \circ B^{a'}$