Lecture 6 - Public Key Cryptography

Gidon Rosalki

2025-12-17

Notice: If you find any mistakes, please open an issue at `https://github.com/robomarvin1501/notes_intro_to_crypto`
We will begin with the concept of key exchange, and move on to real cryptography systems.

1 Private key cryptography

Recall in the previous section of the course, covering symmetric cryptography, given shared **secret** keys, it is possible to securely communicate over an insecure channel.
This means that we need to obtain these shared secret keys, but we cannot send them over this insecure channel. This results in two problems:

1. Key distribution. We can do this by physically meeting, or having trusted messengers, and the like, but then you are left with the problem of sharing a key with a company like Amazon.
2. Key storage: n users need $\binom{n}{2} \approx n^2$ keys, so each user needs to store $n-1$ keys. Since keys need to be stored securely, this becomes a lot of (increasingly expensive) space

1.1 Diffie - Hellman

In a seminal paper by Diffie and Hellman in 1976, called "New Direction in Cryptography", they presented this apparently impossible problem of securely sharing keys over an insecure channel, and how to solve it. This resulted in a radical change, introducing the idea of public key cryptography. It was one of the first steps of moving cryptography out of the private domain (intelligence, militaries, and the like), and into the public one, which all people could use.

The idea of public key encryption is as follows: Both Alice and Bob have a secret key sk, and public key pk. Bob publicises his public key on his homepage, such that everyone can access it. Alice may now communicate with Bob by encrypting with his public key, and he may decrypt with his private key, or mathematically:

$$Dec_{sk}\left(Enc_{pk}\left(m\right)\right) = m$$

This resolves the problem of everyone needing to store everyone else's keys, since they can just check the shared database of size n, that stores all of the public keys.

Diffie and Hellman envisioned three public key primitives:

- Key agreement protocols
- Public key encryption
- Digital signatures

They invented the first key agreement protocol, known as Diffie Hellman key agreement protocol, and the first public-key encryption and digital signature schemes were invented a year later by Rivest, Shamir and Adleman.

1.1.1 Key-Agreement protocols

Alice and Bob run a protocol Π for generating a random key. Note that they are communicating over a potentially unsecured channel, **not** in person. Alice generates r_A, and Bob r_B. $Transcript_{\Pi}\left(1^n, r_A, r_B\right)$ is the transcript of the protocol.

Definition 1.1 (Correctness)**.** Π *is a* ***key agreement protocol*** *if there exists a negligible function* $v\left(n\right)$ *such that for all* $n \in \mathbb{N}$

$$\Pr_{r_A, r_B}\left[K_A\left(1^n, r_A, r_B\right) \neq K_B\left(1^n, r_A, r_B\right)\right] \leq v\left(n\right)$$

This is to say, that K_i generates a different key given the same inputs with an exceedingly low probability. Alice has the inputs $1^n, r_A$, and outputs $K_A\left(1^n, r_A, r_B\right) \in \mathcal{K}_n$. Bob has the same, but with r_B, K_B.
The important thing to note here is that Eve is eavesdropping the communication channel, and should not learn **any** information on the resulting key. Specifically, from Eve's point of view, the key should be "as good as" an independently chosen key.

Definition 1.2 (Security)**.** *A key agreement protocol* Π *is* ***secure*** *if*

$$\left(Transcript_{\Pi}\left(1^{n}, r_{A}, r_{B}\right), K_{A}\left(1^{n}, r_{A}, r_{B}\right)\right) \approx^{c} \left(Transcript_{\Pi}\left(1^{n}, r_{A}, r_{B}\right), K\right)$$

Where $r_A, r_B \leftarrow \{0,1\}^*$, $K \leftarrow \mathcal{K}_n$ *are sampled independently and uniformly.*

In order to create such a protocol, it is important to first remember the definition of *computational indistinguishability*. Two probability distributions are computationally indistinguishable if no efficient algorithm can tell them apart:

Definition 1.3 (Computationally indistinguishable)**.** *Two probability ensembles* $X = \{X_n\}_{n\in\mathbb{N}}, Y = \{Y_n\}_{n\in\mathbb{N}}$ *are* ***computationally indistinguishable*** *if for all PPT distinguishers* $\mathcal{D}$ *there exists a negligible function* $v(\cdot)$ *such that*

$$\left|\Pr\left[\mathcal{D}\left(1^{n}, x\right) = 1\right] - \Pr\left[\mathcal{D}\left(1^{n}, y\right) - 1\right]\right| \leq v\left(n\right)$$

Where $x \leftarrow X_n,\ y \leftarrow Y_n$

This is denoted $X \approx^{c} Y$. We typically consider an efficiently sampleable X and Y. Additionally, from this definition, we can say that pseudorandom means that it is computationally indistinguishable from the uniform distribution.

1.1.2 Diffie-Hellman Key Agreement

Let $\mathcal{G}$ be a PPT algorithm that on input 1^n, outputs $(\mathbb{G}, q, g)$, where $\mathbb{G}$ is a cyclic group of order q, that is generated by g, and q is an $n$bit prime. Let us assume that $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$ is generated, and known to both parties (a publicly published one in the world).

So, Alice samples $x \leftarrow \mathbb{Z}_q$, and then computes $h_A = g^x$, which she sends to Bob. Similarly, Bob samples $y \leftarrow \mathbb{Z}_q$, computes $h_B = g^y$, which he sends to Alice. Alice then outputs $K_A = (h_B)^x$, and Bob outputs $K_B = (h_A)^y$. Here, remember from the earlier definition that $\Pr[K_A \neq K_B] \leq v(n)$ for a negligible v, since as we can see

$$h_A = g^x \tag{1}$$
$$h_B = g^y \tag{2}$$
$$K_A = (h_B)^x = (g^y)^x = (g^x)^y = (h_A)^y = K_B \tag{3}$$

So, above shows correctness, but we need to show security:

$$\left(Transcript_{\Pi}\left(1^{n}, r_{A}, r_{B}\right), K_{A}\left(1^{n}, r_{A}, r_{B}\right)\right) \approx^{c} \left(Transcript_{\Pi}\left(1^{n}, r_{A}, r_{B}\right), K\right)$$

Definition 1.4 (The Decisional Diffie Hellman (DDH) Assumption)**.** *For every PPT algoirithm* $\mathcal{A}$ *there exists a negligible function* $v(\cdot)$ *such that*

$$\left|\Pr\left[\mathcal{A}\left(\mathbb{G}, q, g, g^{x}, g^{y}, g^{xz}\right) = 1\right] - \Pr\left[\mathcal{A}\left(\mathbb{G}, q, g, g^{x}, g^{y}, g^{z}\right) = 1\right]\right| \leq v\left(n\right)$$

Where $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$*, and* $x, y, z \leftarrow \mathbb{Z}_q$

Effectively, they made an assumption that it is secure, and it has still not been broken. If you break it, you will get the Turing prize. Sadly, unlike Computability and Complexity, no guarantees of 100% in the course.

Definition 1.5 (Computational Diffie-Hellman Assumption)**.** *For every PPT algorithm* $\mathcal{A}$*, there exists a negligible function* $v(\cdot)$ *such that*

$$\left|\Pr\left[\mathcal{A}\left(\mathbb{G}, q, g, g^{x}, g^{y}\right) = g^{xy}\right]\right| \leq v\left(n\right)$$

Where $(\mathbb{G}, q, g) \leftarrow \mathcal{G}(1^n)$*, and* $x, y \leftarrow \mathbb{Z}_q$

If you can solve CDH, then you can also solve DDH, so therefore DDH is a more secure assumption.

Some notes: Random elements vs random strings: Alice and Bob agree on a random group element $g^{xy} \in \mathbb{G}$. They typically need a random n bit key $K \in \{0,1\}^n$. There are generic tools to extract such a key (called randomness extractors).

There is also insecurity against active adversaries. Consider if Eve can change what is happening across the channel, then Eve may perform individual key exchange with both Alice, and Bob, and they will be none the wiser, but all their messages will pass through Eve.

Our picture of cryptographic primitives has grown! Behold:

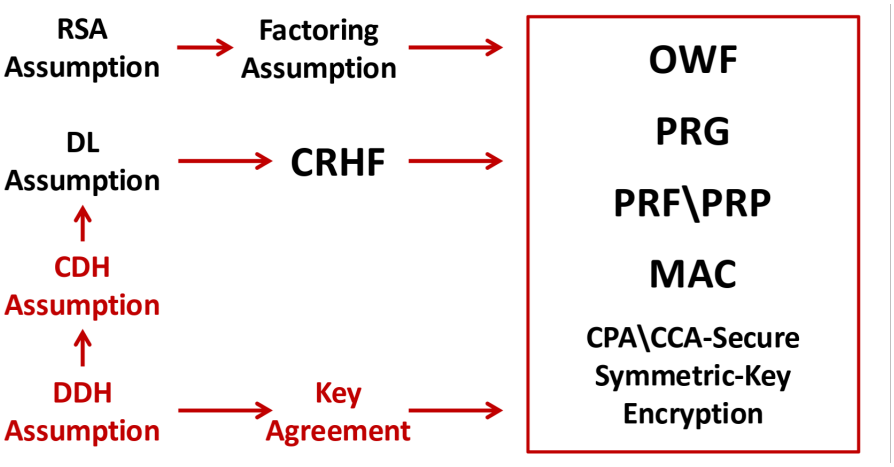

Figure 1: Cryptographic primitives