Discrete Logarithms and the Diffie-Hellman protocol

Reading

- Section 10.4

Practice Problems

10.4 1-5

Notes

Discrete Logarithms

We have seen that when working modulo a prime p there is a primitive root $\bar{a} \in \mathbb{Z}_p$, with the property that any non-zero element is a power of $\bar{a}$, unique modulo $p-1$.

> We call the number $\bar{y} \in \mathbb{Z}_{p-1}$ the **discrete logarithm** of $\bar{x} \in \mathbb{Z}_p$ with base $\bar{a}$ if
>
> $$\bar{a}^y = \bar{x}$$
>
> We write $\bar{y} = \log_{\bar{a}}(\bar{x})$.
>
> We can think of the logarithm by saying that we have a 1-1 and onto function:
>
> $$\mathbb{Z}_{p-1} \longrightarrow \mathbb{Z}_p^*$$
>
> Defined by $\bar{y} \mapsto \bar{a}^{\bar{y}}$. This function is 1-1 and onto, and turns addition into multiplication, and the discrete logarithm is its inverse.

Let's do an example, with $p = 37$. In the previous section we showed that $\bar{2}$ is one of the primitive roots. We will find the logarithms of some elements:

x	log
1	0
2	1
4	2
8	3
16	4
32	5
27	6
17	7
34	8
31	9
25	10

Basically the only efficient way to find the discrete logarithm of a number is to try all the exponents out up to $p-2$, noone has found a faster way.

> Given $\bar{a}$ and $\bar{x}$, there is no efficient way to compute $\log_{\bar{a}}(\bar{x})$.

The Diffie-Hellman protocol

The problem solved by Diffie, Hellman and Merkle is the following:

> **Key exchange problem**
>
> How can two parties agree on a key in such a way that someone intercepting their communications will be unable to determine the value of the key?

Unlike public key cryptography and RSA, where each party provided their own version of a "public key" and a "private key", in this case the goal is to create a **shared private key**. Discrete logarithms are a key step in the process.

At the heart of the process is the following:

- Fast exponentiation allows us to quickly raise a primitive root to any power.
- The reverse process, discrete logarithm, is practically not possible.

Here are the steps in the protocol:

- Alice and Bob agree on a prime p and a primitive root $\bar{a}$ modulo p. Everyone is aware of p, $\bar{a}$.
- Alice randomly chooses a number $1 \leq m \leq p-2$ and computes $\bar{M} = \bar{a}^m$. She transmits $\bar{M}$ to Bob.
- Bob similarly chooses at random a number $1 \leq n \leq p-2$ and computes $\bar{N} = \bar{a}^n$. Bob transmits N to Alice.
- Eavesdroppers can see M, N, but they do not see and cannot compute m, n.
- Both Alice and Bob can compute $\bar{k} = \bar{a}^{mn} = N^m = M^n$. Eavesdroppers cannot.
- Alice finds their secret key by computing N^m.
- Bob finds their secret key by computing M^n.

For a little example, let us revisit our example with $p = 37$ and $\bar{a} = 2$. We need to pick two random numbers between 0 and 35, and let's say we end up with $m = 29$ and $n = 31$.

Alice sees the $m = 29$ and she computes $M = 2^{29} = 2^{10}2^{10}2^9 = 25 \cdot 25 \cdot 31 = 24$. She shares that number with Bob.

Bob sees the $N = 31$ and he computes $N = 2^{31} = (2^{10})^3 2 = 25^3 2 = 22$. He shares that number with Alice.

To find their secret key, Alice would compute $22^{29} = 19$. Bob would instead compute $24^{31} = 19$. 19 is their secret key.