4
Public-Key Cryptography

4.1 The Diffie-Hellman Key Exchange

I recently watched a TV show called *La Femme Nikita* about a skilled women named Nikita, who is forced to be an agent for the anti-terrorist organization Section One. Nikita has strong feelings for fellow agent Michael, and she mosts trust Walter, Section One's gadgets and explosives expert. Often Nikita's worst enemies are her superiors and coworkers at Section One.

The synopsis for a third season episode (which I haven't watched) is as follows:

> PLAYING WITH FIRE
>
> On a mission to secure detonation chips from a terrorist organization's heavily armed base camp, Nikita is captured as a hostage by the enemy. Or so it is made to look. Michael and Nikita have actually created the scenario in order to secretly rendezvous with each other. The ruse works, but when Birkoff [Section One's master hacker] accidentally discovers encrypted messages between Michael and Nikita sent with Walter's help, Birkoff is forced to tell Madeline. Suspecting that Michael and Nikita may be planning a coup d'tat, Operations and Madeline

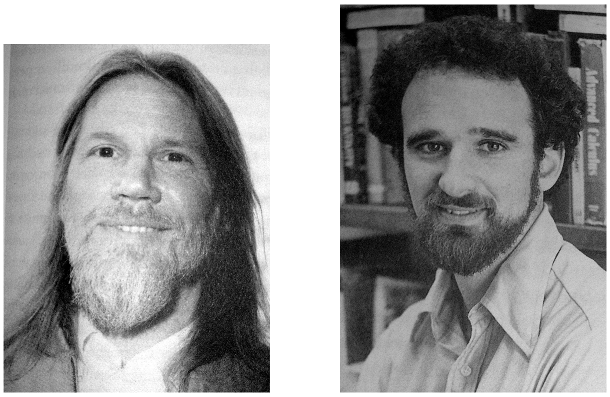


FIGURE 4.1. Diffie and Hellman (photos from [Sin99])

> use a second team of operatives to track Michael and Nikita's next secret rendezvous... killing them if necessary.

What sort of encryption might Walter have helped them to use? I let my imagination run free, and this is what I came up with. After being captured at the base camp, Nikita is given a phone by her captors, in hopes that she'll use it and they'll be able to figure out what she is really up to. Everyone is eagerly listening in on her calls.

Nikita remembers a conversation with Walter about the first public key-exchange protocol, the "Diffie-Hellman key exchange". She remembers that it allows two people to agree on a secret key in the presence of eavesdroppers. Moreover, Walter mentioned that though Diffie-Hellman was the first ever public-key exchange system, it is *still* in common use today (e.g., in `ssh` and `SSL`). It must be good. Perfect!

Nikita pulls out her handheld computer and phone, calls up Michael, and they do the following:

1. Together they choose a big prime number p and a number g with $1 < g < p$.

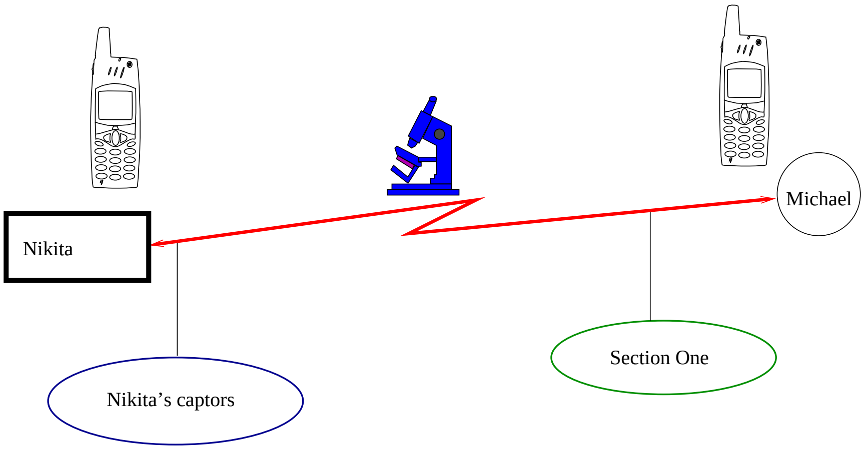
2. Nikita *secretely* chooses an integer n.

3. Michael *secretely* choses an integer m.

4. Nikita tells Michael $ng \pmod{p}$ (the remainder of ng reduced modulo p).

5. Michael tells $mg \pmod{p}$ to Nikita.

6. The "secret key" is $s = nmg \pmod{p}$, which both Nikita and Michael can easily compute.



Here's a very simple example with small numbers that illustrates what Michael and Nikita do. (They really used 200 digit numbers.)

1. $p = 97$, $g = 5$
2. $n = 31$
3. $m = 95$
4. $ng \equiv 58 \pmod{97}$
5. $mg \equiv 87 \pmod{97}$
6. $s = nmg = 78 \pmod{97}$

Nikita and Michael are foiled because everyone easily figures out s:

1. Everyone knows p, g, $ng \pmod{p}$, and $mg \pmod{p}$.
2. Using the very fast Euclidean algorithm, anyone can easily find $a, b \in \mathbb{Z}$ such that $ag + bp = 1$, which exist because $\gcd(g, p) = 1$.
3. Then $ang \equiv n \pmod{p}$, so everyone knows Nikita's secret key n, and hence can find s just as easily as she did.

To taunt her, Nikita's captors give her the Math Review of Diffie and Hellman's 1976 paper "New Directions in Cryptography":

> "The authors discuss some recent results in communications theory [...] The first [method] has the feature that an unauthorized 'eavesdropper' will find it computationally infeasible to decipher the message [...] They propose a couple of techniques for implementing the system, but the reviewer was unconvinced."

Night darkens her cell as Nikita reflects on what has happened. Upon realizing that she misremembered how the system works, she phones Michael and they do the following:

1. Together Michael and Nikita choose a 200-digit (pseudo-)prime p and a number g with $1 < g < p$.

2. Nikita *secretly* chooses an integer n.
3. Michael *secretly* chooses an integer m.
4. Nikita computes $g^n \pmod{p}$ on her handheld computer and tells Michael the resulting number over the phone. (She is surprised that her handheld computer finds $g^n \pmod{p}$ quickly, even though n is very large. How it does this was described in Section 3.5.)
5. Michael tells Nikita $g^m \pmod{p}$.
6. The secret key is then

$$s \equiv (g^n)^m \equiv (g^m)^n \equiv g^{nm} \pmod{p}.$$

Here's a simplified example that illustrates what they did, but which involves only relatively simple arithmetic.

1. $p = 97$, $g = 5$
2. $n = 31$
3. $m = 95$
4. $g^n \equiv 7 \pmod{p}$
5. $g^m \equiv 39 \pmod{p}$
6. $s \equiv (g^n)^m \equiv 14 \pmod{p}$

4.1.1 The Discrete Log Problem

Nikita communicates with Michael by encrypting everything using their agreed upon secret key. In order to understand the conversation, the eavesdroppers need s, but it takes a long time to compute s given only p, g, g^n, and g^m. One way would be to compute n from knowledge of g and g^n; this is possible, but appears to be "computationally infeasible", in the sense that it would take too long to be practical.

Let a, b, and n be real numbers with $a, b > 0$ and $n \geq 0$. Recall that

$$\log_b(a) = n \text{ if and only if } a = b^n.$$

The $\log_b$ function is used in algebra to solve the following problem: Given a base b and a power a of b, find an exponent n such that

$$a = b^n.$$

That is, given $a = b^n$ and b, find n.

Example 4.1.1. The number $a = 19683$ is the nth power of $b = 3$ for some n. With a calculator we quickly find that

$$n = \log_3(19683) = \log(19683)/\log(3) = 9.$$

A calculator computes an approximation for $\log(x)$ quickly by computing a partial sum of a rapidly-converging infinite series.

The discrete log problem is the analogue of this problem but in any finite ("discrete") group:

Problem 4.1.2 (Discrete Log Problem). Let G be a finite group, e.g., $G = (\mathbb{Z}/p)^\times$. Given $b \in G$ and a power a of b, find the smallest positive integer n such that $b^n = a$. Thus the discrete log problem is the problem of computing $n = \log_b(a)$ for $a, b \in G$.

As far as we know, computing discrete logarithms is very time consuming in practice. Over the years, many people have been very motivated to try. For example, if Nikita's captors could efficiently solve Problem 4.1.2, then they could read the messages she exchanges with Michael. Unfortunately, we have no proofs that computing discrete logarithms on a classical computer is difficult. In contrast, Peter Shor [Sho97] showed that quantum computers of significant complexity can solve the discrete logarithm problem in time bounded by a polynomial in the number of digits of $\#G$.

It's easy to give a *slow* algorithm that inefficiently solves the discrete log problem. Simply try b^1, b^2, b^3, etc., until we find an exponent n such that $b^n = a$. For example, suppose $a = 18$, $b = 5$, and $p = 23$. We have

$$b^1 = 5,\ b^2 = 2,\ b^3 = 10,\ \ldots,\ b^{12} = 18,$$

so $n = 12$.

When p is large, computing the discrete log this way soon becomes impractical, because doubling the number of digits of the modulus makes the computation take much longer.

4.1.2 Realistic Example

In this section we present an example that uses bigger numbers.

Let $p = 93450983094850938450983409623$ and $g = -2 \in (\mathbb{Z}/p)^\times$, which has order $p - 1$. The secret random numbers generated by Nikita and Michael are

$$n = 18319922375531859171613379181$$

and

$$m = 82335836243866695680141440300.$$

Nikita sends

$$g^n = 45416776270485369791375944998 \in (\mathbb{Z}/q)^\times$$

to Michael, and Michael sends

$$g^m = 15048074151770884271824225393 \in (\mathbb{Z}/q)^\times$$

to Nikita. They agree on the secret key

$$g^{nm} = 85771409470770521212346739540 \in (\mathbb{Z}/q)^\times.$$

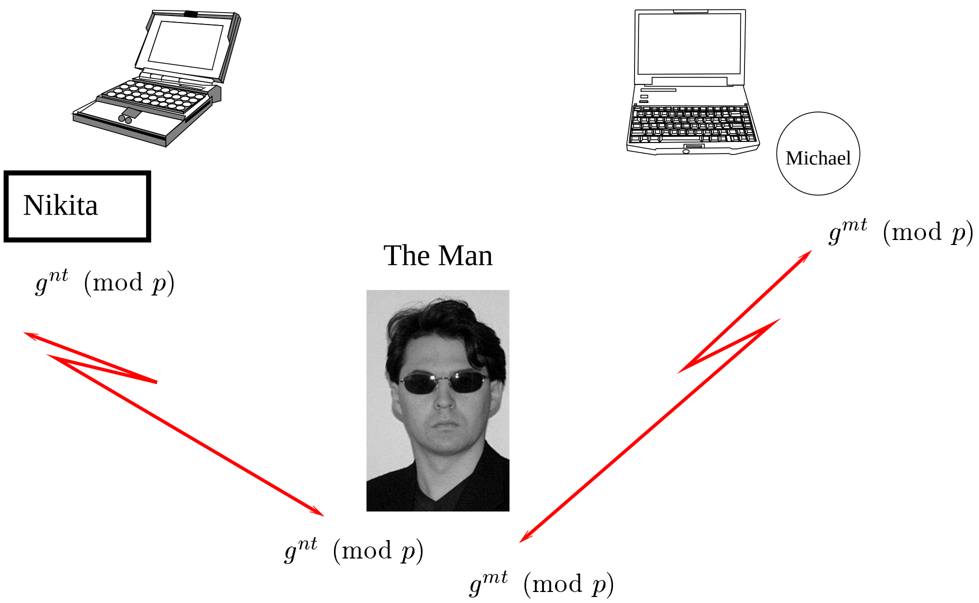


FIGURE 4.2. The Man in the Middle Attack

4.1.3 The Man in the Middle Attack

After their first system was broken, instead of talking on the phone, Michael and Nikita can now only communicate via text messages. Her captor, The Man, is watching each of the email transmissions; moreover, he can intercept messages and send false messages. When Nikita sends an email to Michael announcing $g^n \pmod p$, The Man intercepts this message, and sends his own number $g^t \pmod p$ to Michael. Eventually, Michael and The Man agree on the secret key $g^{tm} \pmod p$, and Nikita and The Man agree on the key $g^{tn} \pmod p$. When Nikita sends a message to Michael she foolishly uses the secret key $g^{tn} \pmod p$; The Man then intercepts it, decrypts it, changes it, and re-encrypts it using the key $g^{tm} \pmod p$, and sends it on to Michael. This is bad.

One way to get around this attack is to use "digital signatures" based on the RSA cryptosystem. We will not discuss digital signatures in this book, but we will discuss RSA in the next section.