6.4 Diffie-Hellman Key Exchange

We can now use modular arithmetic to devise a secure communication protocol. We begin by discussing a method by which two people, far away from one another, can share a password that no one else can know. What is amazing is that both of them can send information publicly, yet end up with a mutually-shared password that only these two people know. How can they do that?

To motivate the general approach, consider the following dilemma. Suppose you and a friend would each like to paint your rooms with the same color. It's not important what color that is, but you want to make sure that no one else in town uses that color. How can you make this happen? If the two of you go to Lowe's or Home Depot together, you can choose a color, split the can, and go home. But suppose that you to live some distance away and won't have a chance to see each other. If one of you buys the paint and sends half of it to the other person, someone else, perhaps the delivery person, might intercept that color! Even if the person only sends the information about the paint, someone else might discover your scheme. Is there any way to solve this problem?

Mixing Paints

It turns out that there is such a way, due to a very important "problem" that arises in mixing paints that some readers have likely encountered. Imagine going to the store and choosing a color you like, and also a bucket of white paint, which you use to make the color lighter. You go home and mix some blue paint and some white until you get the color that you think will look perfect. You begin painting the room but soon, after painting half of the walls, you realize that you didn't mix enough paint, and you'll need to make more. Now you have a mega-problem. You don't remember exactly how much white you added to the blue, and have no idea how to recreate the exact shade you made initially. Of course you can guess the proportions, but now there's a good chance that half of your walls will be one shade of blue, and the other half of the walls will be another shade. You've painted yourself into a figurative corner.

This problem highlights the following beautiful property of paints – it's very easy to mix them, but almost impossible to look at a mixed paint and determine how it was made. While this can be very frustrating for someone painting, this issue in fact allows you and your friend to solve your high-security room-painting needs. You can do the following. Each of you takes a gallon of white paint. Next, you take a colored paint of your choosing in an amount of your choosing and add that to the gallon of white paint you bought; you don't tell anyone how much you've added. Your friend does the same with whatever color they've chosen. Now each of you sends that paint to the other person. The important point to notice is that anyone that might see the paints in transit has no way of knowing what other paint you've mixed in and in what quantity. Perhaps you've added a quarter gallon of quarter gallon of Fountain blue, or perhaps it was a third of a gallon of Capri.

Now you have the paint your friend has mixed, and they have the paint that you've mixed. These paints are different, but you can now make them the same quite easily. Each of you adds to the paint in your hands the exact amount of whatever paint you've chosen and added to the other paint. The two paints are now identical and only the two of you have that color, since anyone in the middle who has seen the paint in transit has no way of determining what color and what amount each of you have added.

This beautiful "thought experiment" shows that it is possible for two people to work together to create information that is known only to them and secret from everyone else, even though they have shared some information publicly. This idea motivates the development of the Diffie-Hellman key-exchange protocol that is used regularly by computers when information must be sent securely. Of course computers do not send paints to one another, but through modular arithmetic they are able to achieve a similar result.

Diffie-Hellman Key Exchange

Alice and Bob would like to communicate securely. The Diffie-Hellman key exchange protocol allows them to work together to create a password that only the two of them will know, even while some of the information they exchange is completely public. To do this, they use numbers instead of paints. More specifically, they agree (publicly) on a modulus m and an integer g, which is smaller than m and which serves as their "white paint". Next, each of Alice and Bob chooses another secret number which they will share with nobody; we will use a to refer to Alice's secret number and b to refer to Bob's secret number.

Alice then calculates $g^a \pmod{m}$ and Bob calculates $g^b \pmod{m}$. Like with the paints, it is easy to create these numbers but almost impossible to figure out how they were made. That is, if you just know g, m, and $g^a \pmod{m}$, there is no known way of efficiently determining a. If m is small we can use trial-and-error to quickly determine a, but in general the value of m might have hundreds of digits (we're talking about numbers bigger than a trillion times a trillion times a trillion many times over). For this reason, Alice can send the number $g^a \pmod{m}$ to Bob and not worry that anyone will figure out her secretly chosen number a, even if they know g, m, and $g^a \pmod{m}$. Likewise, Bob can send over $g^b \pmod{m}$ and not worry that someone will figure out his secret number b. In this sense, they are sending over their specially-mixed paints and no one can figure out how they mixed them, even if they know that the base was white.

At this point Alice still remembers her secret number a and now has a number $g^b \pmod{m}$ which she received from Bob. Using this, and modular exponentiation, she can quickly compute $(g^b)^a \pmod{m}$ by taking $g^b \pmod{m}$ to the a^{th} power. Likewise, Bob still has his secret number b and also knows $g^a \pmod{m}$, which Alice told him, allowing him to compute $(g^a)^b \pmod{m}$. We might remember from high-school algebra that $(x^a)^b = x^{ab} = x^{ba} = (x^b)^a$; the same rules hold in modular arithmetic, and so $(x^a)^b \equiv x^{ab} \equiv x^{ba} \equiv (x^b)^a \pmod{m}$. Therefore, Alice and Bob now have the same number $g^{ab} \pmod{m}$, and they are the only two people that know the number. Even though bad guys might know g and m, and even g^a and $g^b \pmod{m}$, they have no way to figure out $g^{ab} \pmod{m}$.

If Alice and Bob use $g = 3$ and modulus $m = 19$, for example, then if we can just compute $g^1, g^2, \ldots, g^{18}$ to determine all possible values of $g^a \pmod{m}$, and use that list to determine a once we know $g^a \pmod{m}$. As noted above, however, m is usually chosen to be a number with hundreds and hundreds of digits, and calculating a list of possible $g^a \pmod{m}$ values for every $a < m$ would take billions and billions of years, even if we had the most powerful computers in the world focused on that problem alone. It is thus the *practical* impossibility of determining a that makes this protocol secure. We don't know whether one day someone will figure a way to determine a; if that happens, security as we know it will need new tools.

NOTE: In setting up this protocol, it is important to make "good" choices of g and m. To highlight why some thought is necessary, consider choosing $g = 10$ and $m = 101$. We might notice quickly notice that $g^1, g^2, g^3, g^4, g^5 \ldots \equiv 10, 100, 91, 1, 10, \ldots \pmod{101}$. In other words, the repeating pattern has period 4, and so there are only 4 possible values of $g^n \pmod{101}$. That means that Alice and Bob will only be able to send over one of these 4 numbers, making it extremely easy to crack this code.

Example 1. Alice and Bob agree to use $g = 7$ and $m = 997$. Alice chooses $a = 5$ and Bob chooses $b = 10$; they keep these numbers secret, telling no one else about them. Alice then calculates $g^a = 7^5 \equiv 855 \pmod{m}$ and Bob calculates $g^b = 7^{10} \equiv 224 \pmod{m}$. Each sends their computed number to the other, so Alice receives 224 and Bob receives 855. They each then compute g^{ab} (mod 997) by taking their received number and exponentiating it to their secret number. Alice determines that $224^5 \equiv 455 \pmod{997}$ and Bob determines that $855^{10} \equiv 455 \pmod{997}$. Both Alice and Bob now can use the number 455 as a shared secret password.

Example 2. Alice and Bob agree to use a base $g = 37$ and modulus $m =$ 2,305,843,009,213,693,951. Alice chooses $a = 537$ and Bob chooses $b =$ 3024934, which they use to calculate $g^a \pmod{m} \equiv$ 957,141,291,894,918,330 and $g^b \pmod{m} =$ 2,210,741,389,954,762,204. Each sends their computed number to the other, so Alice receives $g^b \pmod{m}$ and Bob receives $g^a \pmod{m}$. They each then compute $g^{ab} \pmod{m}$ by taking their received number and exponentiating it to their secret number. Alice determines $(g^b)^a \pmod{m} =$ 2,305,843,009,213,693,951 and Bob determines that $(g^a)^b \pmod{m}$ is that same number. Alice and Bob now can use this number $g^{ab} \pmod{m}$ as a shared secret password to communicate securely. If a bad guy wanted to guess the secret numbers of Alice and Bob, they might need to perform millions of billions of computations to determine that $a = 537$ or that $b = 3024934$.

Of course calculating these numbers is not easy for us to do by hand, but they are relatively straightforward for a well-programmed computer. This protocol allows computers to communicate with one another through insecure, public channels, yet maintain secrecy of the information that is being transmitted. It is used regularly by millions and billions of electronic device around the world every single day.