6.4 Elliptic Curve Cryptography

In this section we discuss an analogue of Diffie-Hellman that uses an elliptic curve instead of $(\mathbf{Z}/p\mathbf{Z})^*$. The idea to use elliptic curves in cryptography was independently proposed by Neil Koblitz and Victor Miller in the mid 1980s. We then discuss the ElGamal elliptic curve cryptosystem.

6.4.1 Elliptic Curve Analogues of Diffie-Hellman

The Diffie-Hellman key exchange from Section 3.1 works well on an elliptic curve with no serious modification. Michael and Nikita agree on a secret key as follows:

1. Michael and Nikita agree on a prime p, an elliptic curve E over $\mathbf{Z}/p\mathbf{Z}$, and a point $P \in E(\mathbf{Z}/p\mathbf{Z})$.
2. Michael secretly chooses a random m and sends mP.
3. Nikita secretly chooses a random n and sends nP.
4. The secret key is nmP, which both Michael and Nikita can compute.

Presumably, an adversary can not compute nmP without solving the discrete logarithm problem (see Problem 3.1.2 and Section 6.4.3 below) in $E(\mathbf{Z}/p\mathbf{Z})$. For well-chosen E, P, and p experience suggests that the discrete logarithm problem in $E(\mathbf{Z}/p\mathbf{Z})$ is much more difficult than the discrete logarithm problem in $(\mathbf{Z}/p\mathbf{Z})^*$ (see Section 6.4.3 for more on the elliptic curve discrete log problem).

6.4.2 The ElGamal Cryptosystem and Digital Rights Management

This section is about the ElGamal cryptosystem, which works well on an elliptic curves. This section draws on a paper by a computer hacker named Beale Screamer who cracked a “Digital Rights Management” (DRM) system.

The elliptic curve used in the DRM is an elliptic curve over the finite field $k = \mathbf{Z}/p\mathbf{Z}$, where

$$p = 785963102379428822376694789446897396207498568951.$$

In base 16 the number p is

$$\text{89ABCDEF012345672718281831415926141424F7},$$

which includes counting in hexadecimal, and digits of e, π, and $\sqrt{2}$. The elliptic curve E is

$$\begin{aligned} y^2 = x^3 &+ 317689081251325503476317476413827693272746955927x \\ &+ 79052896607878758718120572025718535432100651934. \end{aligned}$$

We have

$$\#E(k) = 785963102379428822376693024881714957612686157429,$$

and the group $E(k)$ is cyclic with generator

$$B = (771507216262649826170648268565579889907769254176,\\ 390157510246556628525279459266514995562533196655).$$

Our heroes Nikita and Michael share digital music when they are not out fighting terrorists. When Nikita installed the DRM software on her computer, it generated a private key

$$n = 670805031139910513517527207693060456300217054473,$$

which it hides in bits and pieces of files. In order for Nikita to play Juno Reactor's latest hit `juno.wma`, her web browser contacts a web site that sells music. After Nikita sends her credit card number, that web site allows Nikita to download a license file that allows her audio player to unlock and play `juno.wma`.

As we will see below, the license file was created using the ElGamal public-key cryptosystem in the group $E(k)$. Nikita can now use her license file to unlock `juno.wma`. However, when she shares both `juno.wma` and the license file with Michael, he is frustrated because even with the license his computer still does not play `juno.wma`. This is because Michael's computer does not know Nikita's computer's private key (the integer n above), so Michael's computer can not decrypt the license file.

We now describe the ElGamal cryptosystem, which lends itself well to implementation in the group $E(\mathbf{Z}/p\mathbf{Z})$. To illustrate ElGamal, we describe how Nikita would set up an ElGamal cryptosystem that anyone could use to encrypt messages for her. Nikita chooses a prime p, an elliptic curve E over $\mathbf{Z}/p\mathbf{Z}$, and a point $B \in E(\mathbf{Z}/p\mathbf{Z})$, and publishes p, E, and B. She also chooses a random integer n, which she keeps secret, and publishes nB. Her public key is the four-tuple (p, E, B, nB).

Suppose Michael wishes to encrypt a message for Nikita. If the message is encoded as an element $P \in E(\mathbf{Z}/p\mathbf{Z})$, Michael computes a random integer r

and the points rB and $P + r(nB)$ on $E(\mathbf{Z}/p\mathbf{Z})$. Then P is encrypted as the pair $(rB, P + r(nB))$. To decrypt the encrypted message, Nikita multiplies rB by her secret key n to find $n(rB) = r(nB)$, then subtracts this from $P + r(nB)$ to obtain

$$P = P + r(nB) - r(nB).$$

We implement this cryptosystem in Section 7.6.3.

Remark 6.4.1. It also make sense to construct an ElGamal cryptosystem in the group $(\mathbf{Z}/p\mathbf{Z})^*$.

Returning out our story, Nikita's license file is an encrypted message to her. It contains the pair of points $(rB, P + r(nB))$, where

$$\begin{aligned} rB = (&1796710032183157463850266557330860449821944246 60,\\ &69783438535968636824930128267514183093517631471 8) \end{aligned}$$

and

$$\begin{aligned} P + r(nB) = (&1378510385482644673726451580930040003436391189 15,\\ &11084858922867622405722923022358081502422487569 9). \end{aligned}$$

When Nikita's computer plays `juno.wma`, it loads the secret key

$$n = 670805031139910513517527207693060456300217054473$$

into memory and computes

$$\begin{aligned} n(rB) = (&3289013935187326375771156506017686810440407157 01,\\ &58694783808781599360135056548878884620388798816 2). \end{aligned}$$

It then subtracts this from $P + r(nB)$ to obtain

$$\begin{aligned} P = (&14489646124220757767,\\ &66933778037328409627489513661819460446969683007 4). \end{aligned}$$

The x-coordinate 14489646124220757767 is the key that unlocks `juno.wma`.

If Nikita knew the private key n that her computer generated, she could compute P herself and unlock `juno.wma` and share her music with Michael. Beale Screamer found a weakness in the implementation of this system that allows Nikita to detetermine n, which is not a huge surprise since n is stored on her computer after all.

6.4.3 The Elliptic Curve Discrete Logarithm Problem

Problem 6.4.2 (Elliptic Curve Discrete Log Problem). Suppose E is an elliptic curve over $\mathbf{Z}/p\mathbf{Z}$ and $P \in E(\mathbf{Z}/p\mathbf{Z})$. Given a multiple Q of P, the *elliptic curve discrete log problem* is to find $n \in \mathbf{Z}$ such that $nP = Q$.

For example, let E be the elliptic curve given by $y^2 = x^3 + x + 1$ over the field $\mathbf{Z}/7\mathbf{Z}$. We have

$$E(\mathbf{Z}/7\mathbf{Z}) = \{\mathcal{O}, (2,2), (0,1), (0,6), (2,5)\}.$$

If $P = (2,2)$ and $Q = (0,6)$, then $3P = Q$, so $n = 3$ is a solution to the discrete logarithm problem.

If $E(\mathbf{Z}/p\mathbf{Z})$ has order p or $p\pm1$ or is a product of reasonably small primes, then there are some methods for attacking the discrete log problem on E, which are beyond the scope of this book. It is thus important to be able to compute $\#E(\mathbf{Z}/p\mathbf{Z})$ efficiently, in order to verify that the elliptic curve one wishes to use for a cryptosystem doesn't have any obvious vulnerabilities. The naive algorithm to compute $\#E(\mathbf{Z}/p\mathbf{Z})$ is to try each value of $x \in \mathbf{Z}/p\mathbf{Z}$ and count how often $x^3 + ax + b$ is a perfect square mod p, but this is of no use when p is large enough to be useful for cryptography. Fortunately, there is an algorithm due to Schoof, Elkies, and Atkin for computing $\#E(\mathbf{Z}/p\mathbf{Z})$ efficiently (polynomial time in the number of digits of p), but this algorithm is beyond the scope of this book.

In Section 3.1.1 we discussed the discrete log problem in $(\mathbf{Z}/p\mathbf{Z})^*$. There are general attacks called "index calculus attacks" on the discrete log problem in $(\mathbf{Z}/p\mathbf{Z})^*$ that are slow, but still faster than the known algorithms for solving the discrete log in a "general" group (one with no extra structure). For most elliptic curves, there is no known analogue of index calculus attacks on the discrete log problem. At present it appears that given p the discrete log problem in $E(\mathbf{Z}/p\mathbf{Z})$ is much harder than the discrete log problem in the multiplicative group $(\mathbf{Z}/p\mathbf{Z})^*$. This suggests that by using an elliptic curve-based cryptosystem instead of one based on $(\mathbf{Z}/p\mathbf{Z})^*$ one gets equivalent security with much smaller numbers, which is one reason why building cryptosystems using elliptic curves is attractive to some cryptographers. For example, Certicom, a company that strongly supports elliptic curve cryptography, claims:

> "[Elliptic curve crypto] devices require less storage, less power, less memory, and less bandwidth than other systems. This allows you to implement cryptography in platforms that are constrained, such as wireless devices, handheld computers, smart cards, and thin-clients. It also provides a big win in situations where efficiency is important."

For an up-to-date list of elliptic curve discrete log challenge problems that Certicom sponsors, see [Cer]. For example, in April 2004 a specific cryptosystem was cracked that was based on an elliptic curve over $\mathbf{Z}/p\mathbf{Z}$, where p has 109 bits. The first unsolved challenge problem involves an elliptic curve over $\mathbf{Z}/p\mathbf{Z}$, where p has 131 bits, and the next challenge after that is one in which p has 163 bits. Certicom claims at [Cer] that the 163-bit challenge problem is computationally infeasible.