11.3 Cryptography

In this section we discuss why an analogue of Diffie-Hellman and RSA for elliptic curves. We then discuss the cryptosystem used in version 2 of the Microsoft Digital Rights Management (MS-DRM) system.

11.3.1 Elliptic Curve Analogues of RSA and Diffie-Hellman

The Diffie-Hellman key exchange from Section 4.1 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 $\mathbb{Z}/p$, and a point $P \in E(\mathbb{Z}/p)$.
2. Michael secretely chooses a random m and sends mP.
3. Nikita secretely chooses a random n and sends nP.
4. The secrete 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 4.1.2) in $E(\mathbb{Z}/p)$. For well-chosen E, P, and p experience suggests that the discrete logarithm problem in $E(\mathbb{Z}/p)$ is much more difficult than the discrete logarithm problem in $(\mathbb{Z}/p)^{\times}$.

There is an analogue for elliptic curves of the RSA cryptosystem of Section 4.2, but the author has never heard of anyone actually using it. Nikita sets up a public key so that Michael and others can send her messages, as follows:

1. Nikita secretely chooses primes p and q, and lets $N = pq$.
2. Nikita chooses an elliptic curve E over $\mathbb{Z}/N$ and considers the group $E(\mathbb{Z}/N)$ (see Section 11.2.5 for the meaning of $E(\mathbb{Z}/N)$).
3. Since Nikita knows p and q, she can use a sophisticated polynomial time algorithm of Schoof, Elkies, and Atkin (see [4]) to compute 4
$$m = \#E(\mathbb{Z}/N) = \#E(\mathbb{Z}/p) \cdot \#E(\mathbb{Z}/q).$$
4. Nikita chooses a random integer e between 1 and $m-1$ that is coprime to m. She lets d be the inverse of e modulo m.
5. To encrypt a message to Nikita, Michael encodes the message as a point $P \in E(\mathbb{Z}/N)$, then sends eP. To decrypt, Nikita computes $d(eP) = (de)P = P$.

This is at best no more secure than RSA, since factoring N breaks the cryptosystem, which probably explains why it is so unpopular.

[4] Give reference.

11.3.2 The ElGamal Cryptosystem and Microsoft Digital Rights Management

In this section we discuss the ElGamal cryptosystem, which is as flexible as RSA, but works well on an elliptic curves. It is used in version 2 of the Microsoft Digital Rights Management (MS-DRM) system.

This section draws on a paper by a hacker named Beale Screamer who cracked version 2 of MS-DRM.

The elliptic curve used in MS-DRM is an elliptic curve over the finite field $k = \mathbb{F}_p$, where

$$p = 785963102379428822376694789446897396207498568951.$$

As Beale Screamer remarks, this modulus has high nerd appeal because in hexadecimal it is

$$89\mathrm{ABCDEF}012345672718281831415926141424\mathrm{F}7,$$

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

$$\begin{aligned} y^2 = x^3 &+ 317689081251325503476317476413827693272746955927x \\ &+ 790528966078787587181205720257185354321006519 34. \end{aligned}$$

We have

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

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

$$\begin{aligned} B = (&771507216262649826170648268565579889907769254176, \\ &390157510246556628525279459266514995562533196655). \end{aligned}$$

Our heros Nikita and Michael love to share digital music when they are not out thwarting terrorists. When Nikita installed Microsoft's content rights management software on her laptop, it generated a private key

$$n = 670805031139910513517527207693060456300217054473,$$

which it hid in bits and pieces of files (e.g., `blackbox.dll`, `v2ks.bla`, and `IndivBox.key`). In order for Nikita to play Juno Reactor's latest hit `juno.wma`, her web browser contacts a Microsoft rights management partner. After Nikita sends her credit card number, the rights management partner sends her a license to 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 laptop still does not play `juno.wma`. This is because Michael's laptop does not know Nikita's laptop's private key (the integer n above), so Michael's laptop can not decrypt the license file.

`juno.wma`

11.3.3 The Elliptic Curve Discrete Logarithm Problem

Definition 11.3.1. If E is an elliptic curve over $\mathbb{F}_p$ and B is a point on E, then the *discrete log problem* on E to the base B is the following problem: given a point $P \in E$ such that $P = mB$ for some m, find an integer n such that $P = nB$.

For example, let E be the elliptic curve given by $y^2 = x^3 + x + 1$ over the field $\mathbb{F}_7$. We have

$$E(\mathbb{F}_7) = \{\mathcal{O}, (2,2), (0,1), (0,6), (2,5)\}.$$

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

When p is large, the discrete logarithm problem on an elliptic curve E over $\mathbb{F}_p$ is conjectured to be "very difficult", except in two special cases: if $\#E(\mathbb{F}_p)$ is "smooth" (i.e., a product of small primes) or E is "supersingular" (i.e., $\#E(\mathbb{F}_p) = p + 1$).[5] The Microsoft curve has neither of these 5
deficiencies, so we expect that the discrete logarithm on that curve is dif-

[5]Find references.

ficult. Beale Screamer does not solve the discrete logarithm on E; this is not how he circumvents MS-DRM.

11.3.4 ElGamal

The only public-key cryptosystem that we have studied so far is the RSA cryptosystem (see Section 4.2), which has no natural analogue in the context of elliptic curves. Fortunately the ElGamal public-key cryptosystem lends itself well to implementation in the group $E(\mathbb{F}_p)$.

MS-DRM uses the ElGamal system. 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 $\mathbb{F}_p$, and a point $B \in E(\mathbb{F}_p)$, 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 $P \in E(\mathbb{F}_p)$, Michael computes a random integer r and the of points rB and $P + r(nB)$ on $E(\mathbb{F}_p)$. 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

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

The following example is based on an example taken from Beale Screamer's paper.

Example 11.3.2. Nikita's license files contains the pair of points $(rB, P + r(nB))$, where

$$\begin{aligned} rB = (&17967100321831574638502665573308604498219442466 0,\\ &69783438535968636824930128267514183093517631471 8) \end{aligned}$$

and

$$\begin{aligned} P + r(nB) = (&13785103854826446737264515809300400034363911891 5,\\ &11084858922867622405722923022358081502422487569 9). \end{aligned}$$

Nikita's laptop sneakily loads the secret key

$$n = 670805031139910513517527207693060456300217054473$$

into memory and computes

$$\begin{aligned} n(rB) = r(nB) = (&328901393518732637577115650601768681044040715701,\\ &586947838087815993601350565488788846203887988162). \end{aligned}$$

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

$$P = (14489646124220757767,\\ 669337780373284096274895136618194604469696830074).$$

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

If Nikita knew the private key n that her laptop generated, she could compute P herself and unlock `juno.wma` and share her music with Michael. Beale Screamer found a weakness in Microsoft's system that let him find n:

> "These secret keys are stored in linked lists ... interspersed with the code in the library. The idea is that they can be read by that library, used internally by that library, and never communicated outside the library. Since the `IndivBox.key` file is shuffled in a random way for each client, these keys would be extremely difficult to extract from the file itself. Fortunately, we don't have to: these keys are part of the object state that is maintained by this library, and since the offset within this object of these secret keys is known, we can let the library itself extract the secret keys! The code for this simply loads up the 'black box' library, has it initialize an instance of the object, and then reads the keys right out of that object. This is clearly a weakness in the code which can be corrected by the DRM software fairly easily, but for now it is the basis of our exploit."

Open Problem 11.3.3. *How can Microsoft store data on Nikita's laptop in such a way that Nikita can not access it, but Nikita's laptop can?*

11.3.5 Why Use Elliptic Curves?

There are several advantages to using elliptic curves in cryptography.

Elliptic curve based cryptosystems with relatively small key sizes are, under certain assumptions, provably as secure as "classical" cryptosystems like RSA with much larger key sizes. And size does matter. According to Dan Boneh of Stanford University, Microsoft may soon use an elliptic curve based cryptosystem during the installation of some of their products. It is unreasonable to ask a user to type in a license key with hundreds of digits; using an elliptic curve system, they can ask the user to type in a much smaller key instead.