COS 533: Advanced Cryptography — Princeton University
Lecture 10 (October 16, 2017)
Lecturer: Mark Zhandry — Scribe: Dylan Altschuler

Notes for Lecture 10

1 Motivation for Elliptic Curves

Diffie-Hellman For exchanging keys publicly, the Diffie-Hellman protocol works well in theory, but there is room for improvement in its efficiency in practice. Let us quickly review how it works: Alice and Bob both have private keys, and they would like to communicate in a public channel so as to have some key at the end of their protocol that is known to both of them but not to any outside listener. The protocol is as follows:

1. Let p be some large prime so that $q = 2p + 1$ is also prime. Let g be some element of $\mathbb{Z}_q^\times$ that has order p.
2. Alice picks an element $a \in^\$ \mathbb{Z}_p$; Bob picks an element $b \in^\$ \mathbb{Z}_p$. They send g^a, g^b to each other respectively.
3. Their new shared private key is $g^{ab} \mod q$, which they can both compute from their hidden keys and from the received messages.

This security of this protocol relies on the hardness of Discrete Log and Decisional Diffie-Hellman. Defining the security parameter as $\lambda = \log(p)$, there are known attacks that run in time:

$$2^{\mathcal{O}(\lambda^{1/3}\log^{2/3}(\lambda))}$$

So, under the standard assumption on physical computation that we only need to worry about at most 2^{128}-time adversaries, we then need a security parameter of $\lambda \approx 3000$. If we wished to be even safer and stop even 2^{256}-time adversaries from running this attack on us, we would need $\lambda \approx 15,000$. However, this means that carrying out our protocol could actually take billions of operations to exponentiate g, which is not ideal. What if there were a replacement for DH that was more secure, thus allowing us to use smaller security parameter?

Abstract Diffie-Hellman What was actually special about the particular group $\mathbb{Z}_q^\times$? We could try picking any group G with any operation $\otimes$; we just need to make sure that the Discrete Log is still hard to solve for $(G, \otimes)$ and that computing g^a, g^b is efficient. First, we give two examples of groups that do NOT work for this:

1. Consider the standard finite additive group $\mathbb{Z}_q^+$. Here, the discrete log is simply modular division, which is easy to compute, especially if we take the generator g to be 1. Thus, the discrete log assumption could not hold for this group.

2. Let f be a OWP (one-way permutation) on $\mathbb{Z}_p^+$ for some prime p, and let multiplication be given by $g \otimes h = f(f^{-1}(g) + f^{-1}(h))$ and let $g = f(1)$. Then, $g^a = f(a)$, so solving the Discrete Log for this group would be equivalent to inverting a OWF, which is assumed hard. But, even computing $g \otimes h$ requires inverting a OWF, so not only can an attacker learn nothing, neither can Alice and Bob!

It turns out that finding a group for the DH protocol for which exponentiation is efficient to compute but hard to invert is difficult. The only known groups are the one used in standard DH and one that we will generate using Elliptic Curves!

2 Elliptic Curves

Background Elliptic curves can usually be put in a canonical form known as *Weierstrass Form*:

$$y^2 = x^3 + ax + b$$

We associate to this form a value called the discriminant:

$$\Delta = -16(4a^3 + 27b^2)$$

Of course, since complex roots appear in conjugates, any cubic has at least 1 real root. If the discriminant is negative, then there is exactly one real root, and if the discriminant is positive, then there are three. If $\Delta = 0$, the curve can have corners, so we generally avoid this case.

When considering an elliptic curve over the reals, we let the curve E be the set of points

$$E := \{(x, \pm\sqrt{x^3 + ax + b}) : x^3 + ax + b \geq 0\}$$

so if the discriminant is positive and there are three roots, then the curve E will actually be disconnected into two parts. One could also consider an Elliptic curve over $\mathbb{C}^2$ by letting x, y be complex variables. Then, the solution set has two (real) dimensions in a space that has four (real) dimensions, so it is a two-surface. In fact, the solution set is homeomorphic (a topological notion that roughly means "equivalent up to continuous deformations") to the torus, colloquially known as the "donut". A nice geometric intuition is that if we were to intersect the two-surface that is the elliptic curve in $\mathbb{C}^2$ with the real plane, the intersection would be exactly E, the elliptic curve over the reals. There is another way to construct a shape homeomorphic to the torus

that will be useful in the future. Tile the plane with parallelograms using integer combinations of two non-parallel vectors u, v; consider the quotient map induced by identifying all the edges of these parallelograms corresponding to u together and identifying all the edges corresponding to v together. Then, the space reduces to some fundamental parallelogram with opposing sides identified, which is of course a torus.

Elliptic Curves as Groups Consider the following group induced by an elliptic curve E: the group elements are points on the curve and one extra element that we call ∞. The group operation $P \times Q$ is defined by the following procedure:

1. **Draw the straight line l from P to Q.** If P and Q are the same point then let l be the tangent line to E at P.
2. **Let R be the third point at which l intersects E.** Since E is a cubic, there can be no more than 3 points of intersection. If P and Q are connected by a vertical line, then we use the convention that the third point of intersection is ∞. In the corner-case that P is a local maxima/minima and so there are only two intersections, then P is essentially an intersection of multiplicity two, so we let $R = P$ (and similarly so for Q).
3. Let $P \times Q := (x_R, -y_R)$, where (x_R, y_R) are the x, y coordinates of R. In words, **flip R over the x-axis; this is our answer**.

While this procedure might seem complicated, it is actually very easy to compute: writing the equation of our line l as $y = rx + s$, we have:

$$r = \frac{y_Q - y_P}{x_Q - x_P}$$
$$s = \frac{x_Q y_P - x_P y_Q}{x_Q - x_P}$$

Then, plugging this into the original equation for the elliptic curve and observing that the intersections of l and E are given exactly by $\{P, Q, R\}$ (so that we know how the equation factors), we can solve for x_R:

$$\begin{aligned} & r^2x^2 + 2rsx + s^2 = y^2 = x^3 + ax + b \\ & \implies x^3 - r^2x^2 + (a - 2rs)x + (b - s^2) = 0 = (x - x_P)(x - x_Q)(x - x_R) \\ & \implies x_R = r^2 - x_P - x_Q \end{aligned}$$

where the last line follows from looking at the coefficient of x^2. Then, we can solve for y_R by plugging x_R into the equation for l given by $y = rx + s$. Thus, we see that calculating $P \times Q$ is quite easy for P, Q in general positions. The corner cases are easy, too, and can be done similarly.

Now, we would like to check that this properly defines an abelian group:

1. Closure follows by construction.
2. Commutativity holds since the line defined by P, Q is the same as the line defined by Q, P.
3. The identity is ∞.
4. The inverse of $P = (x_P, y_P)$ is simply $P^{-1} = (x_P, -y_P)$.
5. The group is associative: one can check by brute-force computation. There is also some geometric intuition: consider the elliptic curve over $\mathbb{C}^2$; the solution set is homeomorphic to the torus, which can be thought about as a parallelogram with opposing sides identified. Under this homeomorphism, the group operation becomes standard complex addition (modulo the quotient relationship on the edges of the torus), and modular arithmetic is associative. Thus, the group operation in $\mathbb{C}^2$ is associative. Since the group generated by the real solutions to the elliptic curve is just a subgroup of this group, it is also associative.

Thus, the group generated by E is a valid commutative group, so it is a candidate for Diffie-Hellman.

Elliptic Curves over Finite Fields In cryptography, we need objects to be discrete to be nicely implementable on the computer. So, we can instead define E over some finite field, e.g. $\mathbb{F}_p$, instead of $\mathbb{R}$. By doing this, we lose geometric intuition, but the group laws still hold. However, there are still some obvious questions about the discrete version of E:

1. **Question**: What is $|E|$? We need to be able to find E with $|E| \geq 2^{128}$ for this to be useful in the DH protocol, so if there are only a few solutions to an elliptic curve over $\mathbb{F}$, we would be in trouble.

   Answer: Hasse's Theorem states:

   $$||E| - (q+1)| \leq 2\sqrt{q}$$

   where $q := |\mathbb{F}|$. So, we are in business: just pick a large finite field.

2. **Question**: How can we find points on the curve?

   Answer: Randomly select $x \in \mathbb{F}$, compute $x^3 + ax + b$ and check if this is a quadratic residue (which can be done efficiently using Jacobi symbols). We expect roughly half the elements of $\mathbb{F}$ to be quadratic residues, so this will quickly find us an element.

3. **Question**: How can we find points on the curve?

   Answer: Randomly select $x \in \mathbb{F}$, compute $x^3 + ax + b$ and check if this is a quadratic residue (which can be done efficiently). We expect roughly half the elements of $\mathbb{F}$ to be quadratic residues, so this will quickly find us an element.

4. **Question**: Can we count the number of points on the curve? Can we learn anything about the group structure of E?

   Answer: Schoof's algorithm lets us count points, and there are advanced theorems that completely characterize the structure of E.

Attacks on Elliptic Curves How hard is the Discrete Log for elliptic curves? A "generic attack" is an attack that just uses group operations and completely ignores additional structure. The brute force attack takes time $\mathcal{O}(|E|) \approx \mathcal{O}(q)$. The "Baby-step Giant-step" algorithm takes $\mathcal{O}(q^{1/2})$. It turns out that this bound is actually tight for generic attacks.

What about non-generic attacks? There is a technique called "pairings" that will not be discussed in this class. One can also use corner cases for certain curves, but this only applies to sparse subsets of elliptic curves. So, if one is careful to avoid using this small set of curves, $\mathcal{O}(q^{1/2})$ is the best attack known. Thus, for security against a 2^{128}-time adversary, we only need use 256 bits (and for a 2^{256}-time adversary, we only need use 512 bits), which is far more efficient that standard Diffie-Hellman. Alice and Bob can compute using tens of millions of operations instead of tens of billions. Thus, both the communication and computation costs are much cheaper when using elliptic curves over standard Diffie-Hellman.