

CS559 Curve Based Cryptography - Prof. Ming-Deh Huang
 Scribe: Iftikhar A Burhanuddin
 burhanud@usc.edu
 Class #4 - January 29, 2002

Administrivia: Scribe notes for class #3 are online at the course web-page: <http://www-rcf.usc.edu/~mdhuang/cs599>

Today's class:

1. Diffie Hellman Key exchange
2. ElGamal protocol
3. Elliptic curve analogues of the above

| Classical version | Elliptic curve version |
|----------------------|-------------------------|
| $G = \mathbf{F}_q^*$ | $G = E(\mathbf{F}_q)$ |
| Diffie Hellman | Elliptic Diffie Hellman |
| ElGamal | Elliptic ElGamal |
| DSA | Elliptic DSA |

Diffie-Hellman Key Exchange: Alice and Bob want to talk to each other and at the end of the exchange they want to share a secret key. Eve is an intruder who eavesdrops on the conversation.

Setup: Steps taken by Alice and Bob in accordance

1. Choose $G = \mathbf{F}_q$, where q is either a large prime or a power of a prime, $q \sim 1024$ bits
2. Pick $g \in \mathbf{F}_q^* = \mathbf{F}_q - \{0\}$ (not necessarily a generator of the group) with the *nice* property that $\text{ord}(g)$ contains a large prime.

Remark: Security depends on hardness of discrete log on $\mathbf{F}_q^* \Rightarrow$ Don't choose \mathbf{F}_q^* where discrete log is easy.

Protocol:

1. Alice chooses (secret) random number a and Bob picks (secret) random number b , where $a, b \in \{1, 2, \dots, \#G\}$. Actually it should be $\text{ord}(g)$ instead of $\#G$ but $\text{ord}(g)$ divides $\#G$...

2. Alice sends g^a to Bob and Bob sends g^b to Alice.
3. Alice computes $(g^b)^a$ and Bob computes $(g^a)^b$. And the shared secret key $K = g^{ab}$

Example: $p = 7, g = 3, a, b \in \{1, 2, \dots, 6\}$. Say Alice picks $a = 6$ and Bob picks $b = 4$. Alice computes $g^a = 3^3 \equiv 6 \pmod{7}$ and sends it to Bob and Bob computes $g^b = 3^4 \equiv 4 \pmod{7}$ and sends it to Alice. Alice and Bob compute K independently:

$$\text{Alice: } (g^b)^a = (4)^3 \equiv 1 \pmod{7}$$

$$\text{Bob: } (g^a)^b = (6)^4 \equiv 1 \pmod{7}.$$

Now they use $K = 1$ as their secret key.

Diffie Hellman Problem - DHP:

Given $\mathbf{F}_q, g \in \mathbf{F}_q^*, g^a, g^b$, to compute g^{ab}

Discrete Logarithm Problem - DLP:

Given $\mathbf{F}_q, g \in \mathbf{F}_q^*, g^a$, to compute a

Eve: $g^a, g^b \implies g^{ab}$? Though Eve can snoop and pick up g^a and g^b , she cannot compute g^{ab} . This is called the *Diffie-Hellman assumption*. If Eve can solve the discrete logarithm problem then she can break the system by computing the discrete of g^a and then using a and g^b to compute g^{ab} the secret key.

Hence we see that the Diffie Hellman Problem can be reduced in polynomial time to the Discrete Logarithm Problem $DHP \leq_p DLP$. It is an open problem whether $DLP \leq_p DHP$ and it is conjectured to be true.

Elliptic Curve Diffie Hellman - ECDH

Alice and Bob want to talk to each other and at the end of the exchange they want to share a secret key. Eve is an intruder who eavesdrops on the conversation.

Setup: Steps taken by Alice and Bob in accordance

1. Choose E and \mathbf{F}_q and use $G = E(\mathbf{F}_q)$, where q is either a large prime or a power of a prime
2. Pick a point $Q \in E(\mathbf{F}_q)$ with the *nice* property that $\text{ord}(Q)$ contains a large prime.

Protocol:

1. Alice chooses (secret) random number a and Bob picks (secret) random number b , where $a, b \in \{1, 2, \dots, \#E(\mathbf{F}_q)\}$. Hasse tells us that $\#E(\mathbf{F}_q) = q + 1 - t$ and $|t| \leq 2\sqrt{q}$.
2. Alice sends aQ to Bob and Bob sends bQ to Alice.
3. Alice computes $a(bQ)$ and Bob computes $b(aQ)$. And the shared secret key is $K = abQ$

Elliptic Curve Diffie Hellman Problem - ECDHP:

Given $E, \mathbf{F}_q, Q \in E(\mathbf{F}_q), aQ, bQ$, to compute abQ

Elliptic Curve Discrete Logarithm Problem - ECDLP:

Given $E, \mathbf{F}_q, Q \in E(\mathbf{F}_q), aQ$, to compute a

Eve: $aQ, bQ \implies abQ$? Though Eve can snoop and pick up aQ and bQ , she cannot compute g^{ab} . This is called the *Elliptic Curve Diffie-Hellman assumption*. If Eve can solve the ECDLP then she can break the system by computing the discrete of aQ and then using a and bQ to compute abQ the secret key.

Remarks:

1. Many people believe that ECDLP is much harder than DLP.
2. We have seen classic and elliptic flavours of the Diffie Hellman Key Exchange protocol. So the next logical question to ask is: can Diffie Hellman be defined over any general finite group? The answer is yes provided that we have a representation for the group G so that group operations are efficient.

ElGamal Cryprosystem: Alice wants to send a message m to Bob over an unsecure channel. Eve is an intruder who eavesdrops on the conversation.

Setup: Steps taken by Alice and Bob in accordance

1. Choose $G = \mathbf{F}_q$, where q is either a large prime or a power of a prime
2. Pick $g \in \mathbf{F}_q^* = \mathbf{F}_q - \{0\}$ (not necessarily a generator of the group)

Protocol:

1. Bob picks a random number $d \in \{1, 2, \dots, \#G\}$ and makes g^d public, in particular known to Alice.
 2. Encryption: Alice generates a random number $k \in \{1, 2, \dots, \#G\}$ called the nonce and sends the 2-tuple $(g^k, m(g^d)^k)$ to Bob, where m is the plaintext message. The nonce is used to obviate attacks by replaying earlier messages.
 3. Decryption: Bob unravels the message m by computing $(mg^{dk})(g^k)^{-d}$
- If Eve solves the DLP then the system is broken.

Elliptic Curve ElGamal Cryprosystem: Alice wants to send a message m to Bob over an unsecure channel. Eve is an intruder who eavesdrops on the conversation.

Setup: Steps taken by Alice and Bob in accordance

1. Choose E and \mathbf{F}_q , where q is either a large prime or a power of a prime
2. Pick $Q \in E(\mathbf{F}_q)$

Protocol:

1. Bob picks a random number $d \in \{1, 2, \dots, \#G\}$ and makes dQ public, in particular known to Alice.
2. Encryption: Alice generates a random number $k \in \{1, 2, \dots, \#G\}$ called the nonce and sends the 2-tuple $(kQ, m + k(dQ))$ to Bob, where m is the plaintext message. Note: Since kdQ is a point on $E(\mathbf{F}_q)$, $m + kdQ$ makes sense only if m is a point on the curve.
3. Decryption: Bob unravels m by computing $(m + kdQ) + (-d)kQ$

If Eve solves the ECDLP then the system is broken.

Remark: We need an encoding scheme which maps the message space ω to points on the curve and vice-versa.

$$\begin{aligned} \omega &\longrightarrow E(\mathbf{F}_q) \\ x &\rightarrow (x, y) \end{aligned}$$

If $(x, y) \notin E(\mathbf{F}_q), \in E(\mathbf{F}_q)$ then we'll have to need upto $\log q + 2 \log q = 3 \log q$ bits to represent $\log q$ bits of the message x on the curve.

ECDLP can also be stated as follows: Given $E(\mathbf{F}_q), S, T \in E(\mathbf{F}_q)$, to compute m , such that $T = mS$, if such an m exists, that is if $E \in \langle S \rangle$

Sub exponential time algorithms exist ($L[\frac{1}{3}]$) for the DLP. On the other hand for the general ECDLP the only *known* technique (giant-step baby-step approach) is a general purpose approach that can be applied to any finite cyclic group and takes time $O(\sqrt{N})$, where N is the order of the group. Though when the trace $t = 0, 1, 2$, which corresponds to the cases $|E(\mathbf{F}_q)| = q + 1$ (supersingular), q (anomalous), $q - 1$ *more efficient* algorithms exist.

Note: $L[\frac{1}{3}] \ll O(\sqrt{N}) \sim L[1]$.

Brief survey of algorithms known for the ECDLP and certicom's answers to "Is there a subexponential-time algorithm for ECDLP?" can be found at <http://www.certicom.com/research/ch2.html>,

Representation of points on E/\mathbf{F}_q :

Do we need $2 \log q$ many bits do we need to write down a point $Q = (x_k, y_k) \in E(\mathbf{F}_q)$?

1. Drop y_k coordinate: This creates ambiguity as it is a 2 : 1 map
2. Drop x_k coordinate: This creates ambiguity as it is a 3 : 1 map
3. Represent Q as $\langle x_k, 0 \rangle$ if $y_k = \{0, 1, \dots, \frac{q-1}{2} - 1\}$ and as $\langle x_k, 1 \rangle$ otherwise. Confusion arises if Q is a 2-torsion point, but we assume it happens rarely.