Overview

Today:
- Discuss homework 6 solutions
- Math needed for discrete-log based cryptography
- Diffie-Hellman and ElGamal
- Elliptic Curves - idea and translation of Diffie-Hellman to ECC

Next:
- Quiz on Thursday (based on HW6 & formal models)
- Graded Homework 2 due on Thursday!
- Read Chapter 11 (skip SHA-512 logic and SHA3 iteration function)
- Project project due in two weeks (April 3) - don’t forget this!

The Discrete Log Problem

For every prime number \( p \), there exists a primitive root (or “generator”) \( g \) such that

\[
g^1, g^2, g^3, ..., g^{p-2}, g^{p-1} \quad \text{(all taken mod } p)\]

are all distinct values (so a permutation of 1, 2, ..., \( p-1 \)).

Example: 3 is a primitive root of 17, with powers:

<table>
<thead>
<tr>
<th>( i )</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
<th>16</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 3 \ mod \ 17 )</td>
<td>3</td>
<td>9</td>
<td>10</td>
<td>13</td>
<td>5</td>
<td>15</td>
<td>11</td>
<td>16</td>
<td>14</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>12</td>
<td>2</td>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>

\( f_{g,p}(i) = g^i \mod p \) is a bijective mapping on \( \{1, ..., p-1\} \)

\( f_{g,p}^{-1}(i) \) is easy to compute (modular powering algorithm)

Inverse, written \( \text{dlog}_{g,p}(x) = f_{g,p}^{-1}(x) \), is believed to be difficult to compute
**Diffie-Hellman Key Exchange (DHE)**

Assume g and p are known, public parameters.

<table>
<thead>
<tr>
<th>Alice</th>
<th>Bob</th>
</tr>
</thead>
<tbody>
<tr>
<td>a ← random value from {1, ..., p-1}</td>
<td>b ← random value from {1, ..., p-1}</td>
</tr>
<tr>
<td>A ← g^a mod p</td>
<td>B ← g^b mod p</td>
</tr>
</tbody>
</table>

Bob sends A to Bob, Alice sends B to Alice.

S_a ← B^a mod p
S_b ← A^b mod p

In the end, Alice’s secret (S_a) is the same as Bob’s secret (S_b):

S_a = B^a = g^{ab} = A^b = S_b

Eavesdropper knows A and B, but to get a or b requires solving the discrete logarithm problem!

---

**ElGamal Encryption**

The idea is simple:

Define “long term key” for one side of Diffie-Hellman

Key Generation (Bob):
- b ← random value from {1, ..., p-1}
- B ← g^b mod p
- (B, g, p) is public key (i.e., encryption key) - b is private key

For Alice to send a message to Bob:
- Get (B, g, p) from Bob
- Pick k ← random value from {1, ..., p-1}
- For message M ∈ {1, ..., p-1}, ciphertext (C_1, C_2) = (g^k mod p, M B^k mod p)

For Bob to decrypt ciphertext (C_1, C_2):
- K ← C_1^b mod p // Same as B^k above
- M ← C_2 / K mod p // Same as original plaintext (see DHE for similarity)

---

**Big Warning!!!**

In ElGamal, only one side can be a long-term key!!!

Serial problems if sender re-uses k!
Abstracting the Problem

There are many sets over which we can define powering.

Example: Can look at powers of $n \times n$ matrices ($A^2, A^3, \text{etc.}$)

Any finite set $S$ with an element $g$ such that $f^g: S \rightarrow S$ is a bijection (where $f(x) = g^x$ for all $x \in S$) is called a cyclic group.

- Very cool math here - see Chapter 5 for more info (optional)

If $f^g$ is easy to compute and $f^{-1}$ is difficult, then can do Diffie-Hellman

“Elliptic Curves” are a mathematical object with this property

- Consequence: Elliptic Curves can use shorter numbers/keys than standard Diffie-Hellman - so faster and less communication required!

Elliptic Curves

The basic idea...

Key ideas:
- Formula with $x$ and $y$ defines a set of points $(x, y)$.
- Formula is quadratic in $y$, cubic in $x$
- Since quadratic in $y$, symmetric around $x$ axis

Define “addition of two points”:
- Draw a line through the two points
- Where else does it hit curve
- 3 places because cubic in $x$
- Reflect around $x$ axis

Elliptic Curves over Finite Fields

General formula for “Elliptic Curves over $\mathbb{Z}_p$” ($p$ is prime):

$E_{a,b}$ is the set of points $(x, y)$ satisfying $y^2 \equiv x^3 + ax + b \pmod{p}$

Technical requirement for $a$ and $b$: $4a^3 + 27b^2 \equiv 0 \pmod{p}$

<table>
<thead>
<tr>
<th>$x$</th>
<th>$y^2$ in $E_{a,b}$ (mod 5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$y^2 = x^3 + 2x + 1$ mod 5</td>
</tr>
<tr>
<td>1</td>
<td>$y^2 = x^3 + 2x + 1$ mod 5</td>
</tr>
<tr>
<td>2</td>
<td>$y^2 = x^3 + 2x + 1$ mod 5</td>
</tr>
<tr>
<td>3</td>
<td>$y^2 = x^3 + 2x + 1$ mod 5</td>
</tr>
<tr>
<td>4</td>
<td>$y^2 = x^3 + 2x + 1$ mod 5</td>
</tr>
</tbody>
</table>

Points in $E_{a,b}$

- $O$: $(0,1)$
- $(0,4)$
- $(1,2)$
- $(1,3)$
- $(3,2)$
- $(3,3)$
Elliptic Curves over Finite Fields

General formula for "Elliptic Curves over \( \mathbb{Z}_p \) (p is prime):

\[ E_p(a,b) \text{ is the set of points } (x,y) \text{ satisfying } y^2 \equiv x^3+ax+b \pmod{p} \]

Technical requirement for \( a \) and \( b \): \( 4a^3+27b^2 \not\equiv 0 \pmod{p} \)

Important points

- Can add points as before (no sensible picture, however)
- For a point \( P \), can calculate
  - \( 2P = P+P \)
  - \( 3P = P+P+P \)
  - \( 4P = P+P+P+P \)
  - \( \cdots \) (eventually repeats → \( P \) generates a cyclic group)
- Notation is multiplying rather than powering, but can do Diffie-Hellman!
- Important: Discrete logs seem to be harder to compute for Elliptic Curves than \( \mathbb{Z}_p \)
- Consequence: Elliptic Curves can use shorter numbers/keys than standard Diffie-Hellman - so faster and less communication required!

Revisiting Key Sizes

From NIST publication 800-57a

<table>
<thead>
<tr>
<th>Security Strength</th>
<th>Symmetric key algorithms</th>
<th>ECC (e.g., P192, P256)</th>
<th>BIC (e.g., RSA)</th>
<th>ECC (e.g., ECDSA)</th>
</tr>
</thead>
<tbody>
<tr>
<td>192</td>
<td>AES-256</td>
<td>( 2^{80} )</td>
<td>( 2^{192} )</td>
<td>( 2^{256} )</td>
</tr>
<tr>
<td>128</td>
<td>AES-192</td>
<td>( 2^{56} )</td>
<td>( 2^{160} )</td>
<td>( 2^{192} )</td>
</tr>
<tr>
<td>128</td>
<td>AES-128</td>
<td>( 2^{48} )</td>
<td>( 2^{128} )</td>
<td>( 2^{192} )</td>
</tr>
<tr>
<td>94</td>
<td>AES-94</td>
<td>( 2^{36} )</td>
<td>( 2^{94} )</td>
<td>( 2^{94} )</td>
</tr>
<tr>
<td>76</td>
<td>AES-76</td>
<td>( 2^{16} )</td>
<td>( 2^{76} )</td>
<td>( 2^{76} )</td>
</tr>
</tbody>
</table>

Issue: PK algorithms based on mathematical relationships, and can be broken with algorithms that are faster than brute force.

We spent time getting a feel for how big symmetric cipher keys needed to be

How big do keys in a public key system need to be?

From NIST pub 800-57a: