Massachusetts Institute of Technology
6.857: Network and Computer Security
Professor Ronald L. Rivest

Handout 9
October 2, 1997

Problem Set 4

This problem set is due on *Thursday, October 9, 1997* at the end of class. Late homeworks will *not* be accepted.

Mark the top of each sheet with your name, 6.857, the problem set number, and the date. Type up your solutions, and be clear. Points may be deducted if your TA has problems understanding your solution.

If you collaborate with other students, you MUST write up solutions on your own and acknowledge the people you work with.

Problem 4-1. Elliptic Curve Groups

(a) List all of the distinct elliptic curves modulo 5. Use the Weierstrauss normal form:

$$y^2 = x^3 + ax + b \pmod 5$$

So the problem reduces to finding all of the coefficients a and b such that:

$$4a^3 + 27b^2 \neq 0 \pmod 5$$

For the particular curve:

$$y^2 = x^3 + x + 1 \pmod 5$$

(b) List all of the points on the curve (including the point at infinitiy). **Hint:** there should be 9 points total.

(c) Give the addition table for this group. This is a 9×9 table showing the sum of any two elements. You can use shorthand, e.g. "24" to indicate the point $(x, y) = (2, 4)$ which happens to be on the curve. Give row and column labels in increasing order by this shorthand, with the point I at infinity last.

(d) Give the order of each element in the group.

Hint: Addition. The following is how one can compute the addition of two points in an elliptic curve field $y^2 = x^3 + ax + b$:

- if $(x_1 = x_2)$ and $(y_1 = -y_2)$ then **return**(I).
- **if** $(x_1 = x_2)$ then $\lambda = \frac{3x_1^2 + a}{2y_1}$ **else** $\lambda = \frac{y_2 - y_1}{x_2 - x_1}$
- $\beta = y_1 - \lambda x_1$

- $x_s = \lambda^2 - x_1 - x_2$
- $y_s = -(\lambda x_s + \beta)$
- **return** (x_s, y_s)

Problem 4-2. Hash Functions

Definition 1 *A hash function is* **weakly collision-free** *if, given a message x, it is computationally infeasible to find a message $x' \neq x$ such that $h(x') = h(x)$.*

Definition 2 *A hash function is* **strongly collision-free** *if it is computationally infeasible to find messages x and x' such that $x' \neq x$ and $h(x') = h(x)$.*

Suppose $h_1 : (\mathbb{Z}_2)^{2m} \to (\mathbb{Z}_2)^m$ is a strongly collision-free hash function.

(a) Define $h_2 : (\mathbb{Z}_2)^{4m} \to (\mathbb{Z}_2)^m$ thus:

- write $x \in (\mathbb{Z}_2)^{4m}$ as $x = x_1 \| x_2$, where $x_1, x_2 \in (\mathbb{Z}_2)^{2m}$
- define $h_2(x) = h_1(h_1(x_1) \| h_1(x_2))$.

Prove that h_2 is strongly collision-free.

(b) For an integer $i \geq 2$, define a hash function $h_i : (\mathbb{Z}_2)^{2^i m} \to (\mathbb{Z}_2)^m$ recursively from h_{i-1} thus:

- write $x \in (\mathbb{Z}_2)^{2^i m}$ as $x = x_1 \| x_2$, where $x_1, x_2 \in (\mathbb{Z}_2)^{2^{i-1} m}$
- define $h_i(x) = h_1(h_{i-1}(x_1) \| h_{i-1}(x_2))$.

Prove that h_i is strongly collision-free.