CS 161: Computer Security

Lecture 4

September 10, 2015

Where we are

- How did NSA break SSL?
- Basic number theory
- RSA
- Digital certificates
- Shamir secret sharing
- Rabin signatures
- Secure hashing
- Elliptic curve cryptography
- Pseudo-random number generation
- SSL protocol

Review: Hash functions

- You remember hash functions from 61B
- Properties
  - Variable input size
  - Fixed output size (e.g., 512 bits)
  - Efficient to compute
  - Psuedo-random (mixes up input well!)
- In this lecture $H()$ denotes a hash function

Review: Probability of a collision

- Suppose hash value range is n
- And k input points are hashed

- Probability of a collision is

$$P(n,k) = 1 - \frac{n!}{(n-k)!\,n^k} \approx 1 - e^{-k^2/2n}$$

Review: Cryptographic hash functions

- Cryptographic hash functions add conditions
- Preimage resistance
  - Given h, intractable to find y such that $H(y) = h$
- Second preimage resistance
  - Given x, intractable to find $y \neq x$ such that $H(y) = H(x)$
- Collision resistance
  - Intractable to find $(x, y), y \neq x$ such that $H(y) = H(x)$

Review: RSA signature

- For large documents m
- Compute $H(m)$
- Sign $H(m)$
- Transmit $\langle m, Sign(H(m)) \rangle$

- This is used in digital certificates (used in SSL)

Review: Modular division mod n

- How to calculate $x^{-1} \bmod n$ (n composite)?
  - Note x, n must be relatively prime
  - Dividing by a modular factor like dividing by zero

- Method 1:
  - Use Extended GCD to solve
  - $ax + bn = 1$
  - $ax \equiv 1 \pmod{n}$ so $a \equiv x^{-1} \pmod{n}$

Review: Modular division

- How to calculate $x^{-1} \bmod n$ (n composite)?
  - Note x, n must be relatively prime
  - Dividing by a modular factor like dividing by zero

- Method 2:
  - Use Fermat-Euler theorem to solve
  - $x^{\varphi(n)} \equiv 1 \pmod{n}$
  - $x^{\varphi(n)-1}x \equiv x^{\varphi(n)} \equiv 1 \pmod{n}$
    so $x^{\varphi(n)-1} \equiv x^{-1} \pmod{n}$

This lecture

- Discrete logarithm problem
- Diffie-Hellman key exchange
- Man-in-the-middle attacks
- Elgamal cryptosystem
- Introduction to elliptic curve cryptography

Generators [primitive roots]

- For prime p, consider $\mathbb{Z}_p$:
  - Integers mod p excepting 0: $\{1, 2, \ldots, (p-1)\}$
- $g \in \mathbb{Z}_p$ is a *generator* if

$$\left\{g^1, g^2, \ldots, g^{(p-1)}\right\} \pmod{p} = \mathbb{Z}_p$$

- Example: 2 is a generator mod 5
  - $\{2^1, 2^2, 2^3, 2^4\} = \{2, 4, 3, 1\} = \{1, 2, 3, 4\} \pmod{5}$
- Example: 4 is <u>not</u> a generator mod 5
  - $\{4^1, 4^2, 4^3, 4^4\} = \{4, 1, 4, 1\} \neq \{1, 2, 3, 4\} \pmod{5}$

Finding a generator g

- To find a generator g in $\mathbb{Z}_p$, factor $p - 1$
  - For each factor p_i of $p - 1$, check
  - $g^{(p-1)/p_i} \neq 1 \pmod{p}$
- Normally it is hard to factor $p - 1$
- We choose p to be of form $2q + 1$
  - q is prime
  - Factors of $p - 1$ are $2, q$

More details: *Handbook of Applied Cryptography*, chap 4

Linked from Piazza

Discrete logarithm problem

- We fix a prime p and a generator $g \in \mathbb{Z}_p$

- Discrete logarithm problem:

Given $a \in \mathbb{Z}_p$, find k such that $g^k \equiv a \pmod{p}$

Discrete logarithm problem hard?

- Best algorithm for factoring: number field sieve
- Best algorithm for discrete log: function field sieve
- In both cases, complexity is $e^{((c+o(1))(\log n)^{1/3}(\log\log n)^{2/3})}$
- 2013, French team found better discrete log algorithm for some special cases
- Quantum computing solves both factoring & discrete log in polynomial time

Will discrete log & factoring be broken soon?

- Rapid algorithmic development
- Some people think that these problems may be unsuitable for crypto
- If so, we need to fall back on elliptic crypto

Diffie-Hellman key exchange

Alice

Bob

prime p, generator $g \in \mathbb{Z}_p$ $\longrightarrow$

$g^A \bmod p$ $\longrightarrow$

$g^B \bmod p$ $\longleftarrow$

$(g^B)^A \mod p$

$(g^A)^B \mod p$

Key exchange

- Now Alice and Bob both have $g^{AB} \bmod p$
- They can use as a secret (shared) key
  - (this is symmetric crypto)
  - (we will study in about a week – AES, DES, etc.)

Man in the middle attack

Alice MITM Bob

$$g^A \bmod p \quad \longrightarrow \qquad\qquad g^S \bmod p \quad \longrightarrow$$

$$g^T \bmod p \quad \longleftarrow \qquad\qquad g^B \bmod p \quad \longleftarrow$$

$$g^{AT} \bmod p \qquad g^{AT}, g^{SB} \bmod p \qquad g^{SB} \bmod p$$

Encrypted channel

Encrypted channel

Elgamal cryptosystem

- Referee
  - prime p, generator g
- Bob
  - random $x \in \{1, 2, \dots, (p-2)\}$
  - $y = g^x \pmod{p}$
  - public key (p, g, y); secret key x
- Alice
  - message M, random $k \in \{1, 2, \dots, (p-2)\}$
  - $a = g^k; b = My^k \pmod{p}$
  - transmits $\langle a, b \rangle$
- Bob
  - $b(a^x)^{-1} = My^k(g^{kx})^{-1} = M(g^x)^k g^{-xk} = M \pmod{p}$

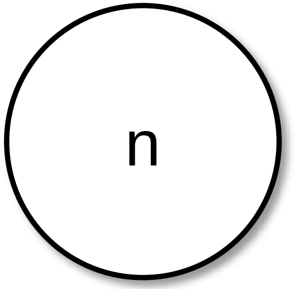
What I learned from Star Trek

- The universe is full of humanoids
- They mostly speak American English
- Computers are evil

- Where is the diversity?

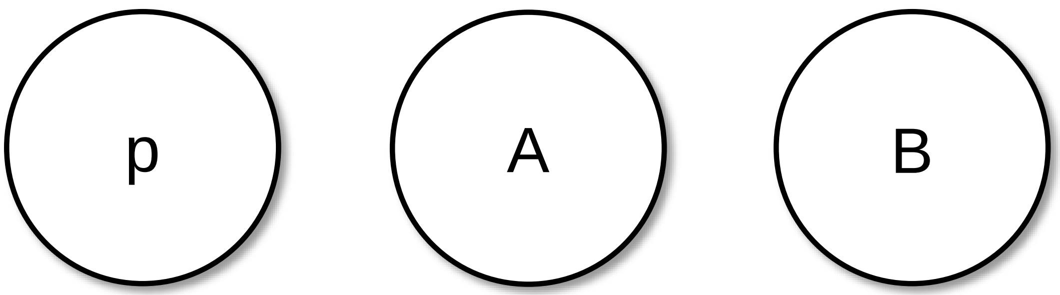
Lack of diversity problem for numbers too!

- Only one sequence of positive integers
  - 1, 2, 3, 4, 5, 6, …
- Maybe factoring (or other problems) easy for integers
- We only have one knob



Elliptic curves add diversity

- We are no longer restricted to a single knob
- Many different “universes” of “numbers”

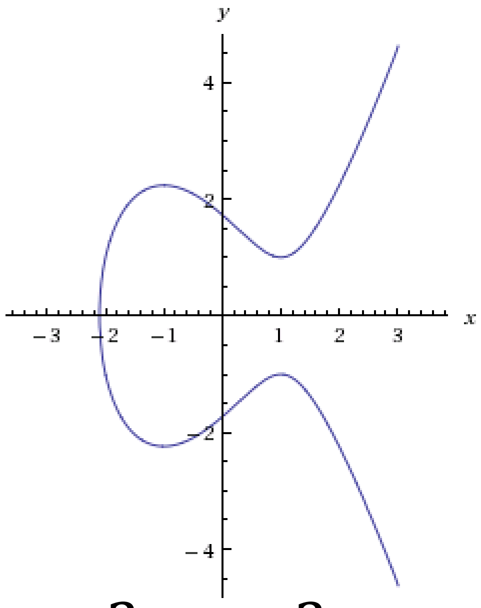


Elliptic curves

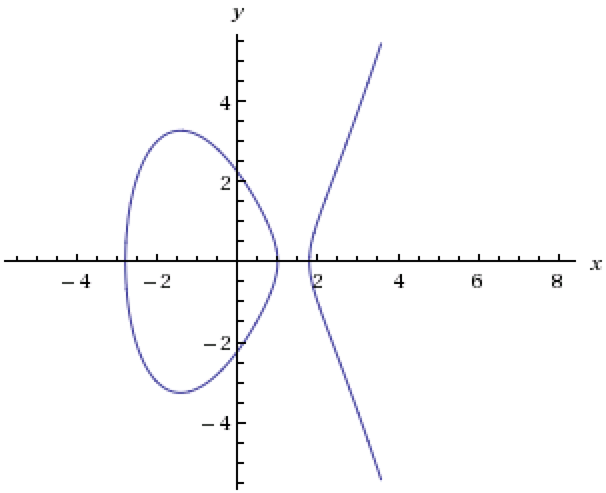
- Many uses: Fermat's Last Theorem, primality testing, factoring
- Physics
  - Exact solution to the pendulum problem
  - Motion of strings in string theory
- Birch-Swinnerton-Dyer conjecture
  - Millennium problem - $1 million prize
- We will use for crypto only
- Elliptic curves are the most commonly used public key cryptosystem

Elliptic curves

- Weierstrass equations
- $y^2 = x^3 + Ax + B$

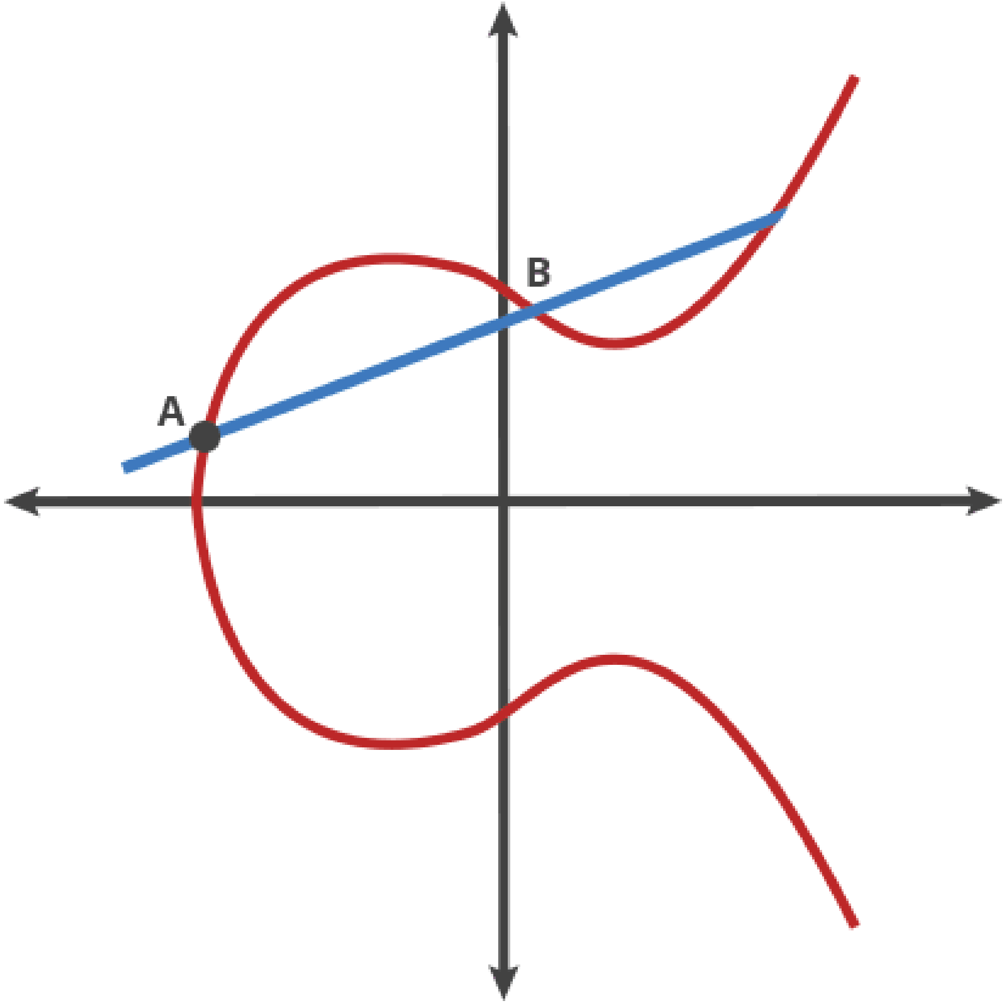


$y^2 = x^3 - 3x + 3$



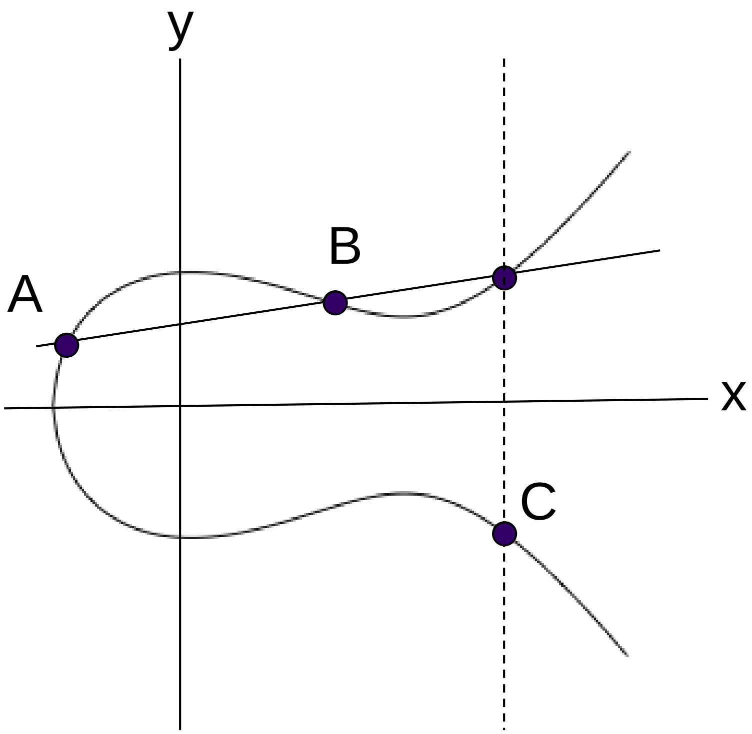
$y^2 = x^3 \ - 6x + 5$

Arithmetic on elliptic curves



$A \oplus B = C \qquad A \oplus C = D \qquad A \oplus D = E$

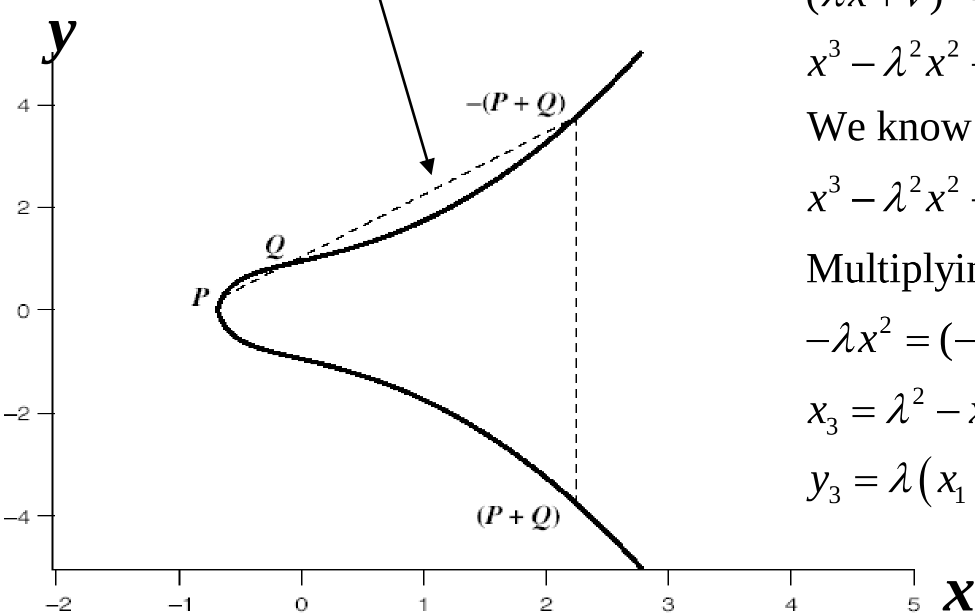
Elliptic Curve operation: $\oplus$



$$C = A \oplus B$$

Elliptic Curve operation: $\oplus$

$$y = \lambda(x - x_1) + y_1$$



$$y^2 = x^3 + Ax + B$$

$$P = (x_1, y_1), Q = (x_2, y_2)$$

$$R = (P \oplus Q) = (x_3, y_3)$$

$$P \neq Q$$

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$$

$$(\lambda x + \nu)^2 = x^3 + Ax + B$$

$$x^3 - \lambda^2 x^2 + (A - 2\lambda\nu)x + (B - \nu^2) = 0$$

We know this is a cubic with factors x_1, x_2, x_3

$$x^3 - \lambda^2 x^2 + (A - 2\lambda\nu)x + (B - \nu^2) = (x - x_1)(x - x_2)(x - x_3)$$

Multiplying out and just taking the x^2 term

$$-\lambda x^2 = (-x_1 - x_2 - x_3)x^2$$

$$x_3 = \lambda^2 - x_1 - x_2$$

$$y_3 = \lambda\left(x_1 - x_3\right) - y_1$$

P⊕P

- $P \oplus P$ requires computing the tangent line

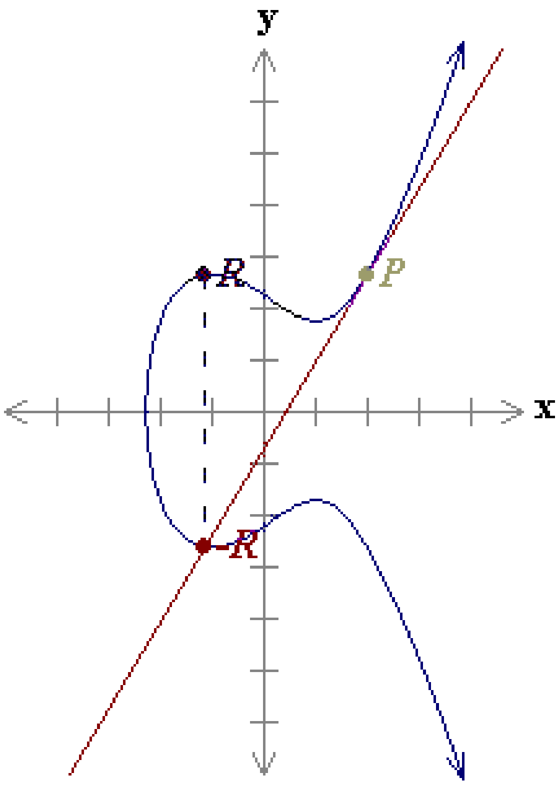
$$y^2 = x^3 + Ax + B$$

Implicit differentiation

$$2y\frac{dy}{dx} = 3x^2 + A$$

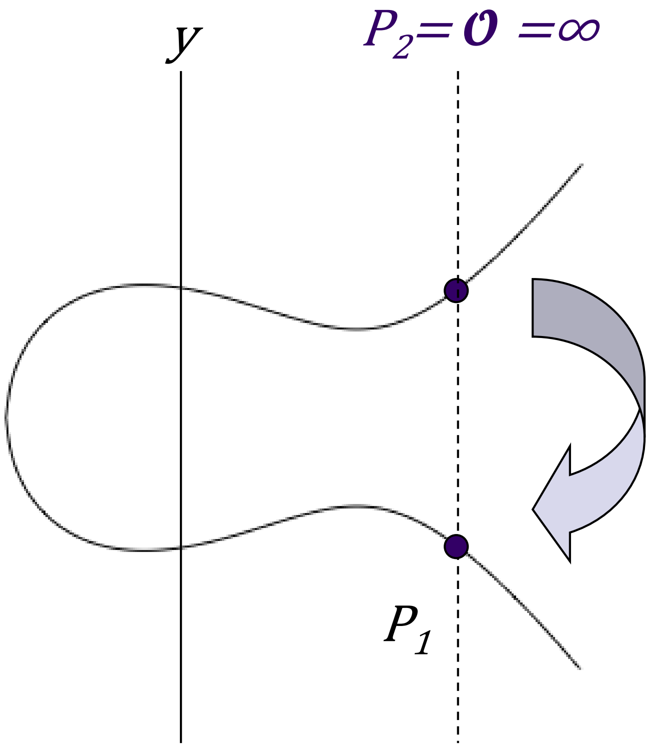
$$\lambda = \frac{dy}{dx} = \frac{3x_1^2 + A}{2y_1}$$

$$x_3 = \lambda^2 - 2x_1, y_3 = \lambda(x_1 - x_3) - y_1$$



$y^2 = x^3 - 3x + 5$

Why do we need the reflection?



$$P_1 = P_1 \oplus \mathcal{O} = P_1$$

Addition rules

- $P \oplus \mathcal{O} = P$
- $(x, y) \oplus (x, -y) = \mathcal{O}$
- $\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1} & \text{if } P \neq Q \\ \dfrac{3x_1^2 + A}{2y_1} & \text{if } P = Q \end{cases}$
- $P \oplus Q = (x_3, y_3)$
- $x_3 = (\lambda^2 - x_1 - x_2)$ & $y_3 = \lambda(x_1 - x_3) - y_1$

Scalar multiplication

- $0P = \mathcal{O}$
- $1P = P$
- $2P = P \oplus P$
- $3P = P \oplus P \oplus P$
- $4P = P \oplus P \oplus P \oplus P$
- …

Next lecture

- Special lecture on Tuesday
  - David Fifield on C
- Lecture on Thursday
  - Pseudo-random number generation
  - SSL