Cryptography

Yesterday: RSA.

How does RSA work?

Based off of being able to compute roots mod m.

i.e. $x^k \equiv b \mod m$ for x.

RSA relies on factoring being hard.

- Pick two large primes p, q. (private)
  $m = pq$.
- Choose $e \in \mathbb{Z}$ w/ $(e, \varphi(m)) = 1$.
  e is called the encryption key

The pair (m, e) form the <u>public key</u>.

To encrypt a message x:

$x \bmod m \rightsquigarrow x^e \bmod m$

To decrypt this message:

find d s.t. $ed \equiv 1 \bmod \varphi(m)$

$y \bmod m \rightsquigarrow y^d \bmod m.$

Why does this work?

$x \bmod m \rightsquigarrow x^e \bmod m$

$\rightsquigarrow (x^e)^d \equiv x^{ed} \equiv x \bmod m.$

RSA is only secure when d is unknown! In practice, this means when $\varphi(n)$ is hard to compute.

An example.

Use a substitution cipher:

A	B	...	Z	Space
01	02		26	27

"MATH IS FUN"

$\downarrow$

1301 2008 | 0727 09 | 1927 06 | 2114

$m = 1223 \cdot 2309 = 2823907$

$e = 3$

Public key is $(2823907, 3)$.

m has 7 digits, so break our message up into blocks of at most 6 digits.

Message:

$(130120, 072709, 192706, 2114)$

Now encrypt:

$130120^3 \bmod 2823907 \equiv 1645568$

$072709^3 \bmod \text{--} \equiv 1228979$

$192706^3 \bmod \cdots \equiv 1307269$

$2114^3 \bmod \cdots \equiv 1488629$

$(1665568, 1228979, 1307269, 1488629)$

Encrypted message.

<u>Ex</u>: Now suppose we receive the following message:

$(1596211, 772934, 2761338, 787573, 1500275, 637504, 993841)$

Public key same as before.

We know $m = 1223 \cdot 2309$

$\varphi(m) = 2820376$

To find d, Solve

$ed \equiv 1 \mod \varphi(m)$

$3d \equiv 1 \mod 2820376.$

$d \equiv 1880251 \mod 2820376$

$d = 1880251.$

To decrypt: raise each block to d mod m: e.g.

$$1596211^{1880251} \mod 2823907$$

$$\equiv 172101.$$

Do this for each block.

Decrypted message:

(172101, 180114, 200914, 52709, 192702, 151809, 1407)

Use cipher to translate back to text:

"QUARANTINE IS BORING."

For a quantum computer, there is an efficient factoring algorithm called Shor's algorithm, makes RSA easy to crack.

Nowadays, many modern methods rely on discrete log problem.

(e.g. Elliptic curve cryptography)

Discrete log

Powers: $x^k \equiv b \mod m$

Exponential: $a^x \equiv b \mod m$

When does a solution to

$a^x \equiv b \mod m$ exist?

A: (for m a prime)

Always!

Why?

<u>Thm:</u> For an odd prime p there is a number g s.t.

$\mathrm{ord}_p(g) = p-1$.

I.e. $g^{p-1} \equiv 1 \mod p$

and $g^k \not\equiv 1 \mod p$ for

$1 \le k \le p-2$.

Proof in lecture at some point in the future (?).

In particular, the thm says

$g, g^2, \ldots, g^{p-2}, g^{p-1} \mod p$

are all distinct. Mod p there are $p-1$ non-zero things, So this hits all of them. Thus,

For any b, $g^k \equiv b \mod m$

for some k.

$a^x \equiv b \mod m$

$a = g^k$

$g^{kx} \equiv b \mod m.$

So only need to solve for base g.

A number g s.t. $\mathrm{ord}_p(g) = p-1$ is called a <u>generator</u> mod p.

$g^x \equiv b \bmod m$ "$x = \log_g(b)$"

<u>Example</u>

Suppose we want to solve

$2^k \equiv 50 \bmod 101.$

Fact: 2 is a generator mod 101.

i.e. $\mathrm{ord}_{101}(2) = 100.$

$100 = 4 \cdot 25 = 2^2 \cdot 5^2.$

Idea: work mod 4, 25
and glue w/ CRT.

Let's start mod 2:

$k \equiv 0 \bmod 2$ or $k \equiv 1 \bmod 2$

If
$k = 2\ell$

$2^k \equiv 2^{2\ell} \equiv 50 \bmod 101$

$\Rightarrow 2^{100\ell} \equiv (2^{100})^\ell \equiv 1$

$\Rightarrow \quad 50^{50} \equiv 1 \mod 101$

$\Rightarrow\!\!\!\Leftarrow \qquad (50^{50} \equiv -1 \mod 101)$

So $k \equiv 1 \mod 2$.

Now work mod 4.

$k = 1+2l$ and l is even or odd,
so

$k \equiv 1 \mod 4$ or $k \equiv 3 \mod 4$

Suppose $k \equiv 3 \mod 4$.

$k = 3+4l$

$2 \cdot 2^k = 2^{k+1} = 2^{4(1+l)}$

$$\Rightarrow (2\cdot 2^k)^{25} \equiv R^{100(1+l)} \equiv 1 \mod 101$$

$$\Rightarrow (2\cdot 50)^{25} \equiv 1 \mod 101$$

$$\Rightarrow\Leftarrow.$$

$$\left((2\cdot 50)^{25} \equiv -1 \mod 101 \right)$$

$$\Rightarrow$$

$k \equiv 3 \mod 4.$

Now work mod 5:

$k \equiv 0 \mod 5 \quad k = 5l \Rightarrow 50^{20} \equiv 1 \Rightarrow\Leftarrow$

$k \equiv 1 \mod 5 \quad k = 1+5l \Rightarrow (2^1\cdot 50)^{20} \equiv 1 \Rightarrow\Leftarrow$

$k \equiv 2 \mod 5 \quad k = 2+5l \Rightarrow (2^2\cdot 50)^{20} \equiv 1 \Rightarrow\Leftarrow$

$k \equiv 3 \mod 5 \quad k = 3+5l \Rightarrow (2^3\cdot 50)^{20} \equiv 1 \Rightarrow\Leftarrow$

$K \equiv 4 \mod 5$ $\quad K = 4+5l$

So

$K \equiv 4 \mod 5$

mod 25:

$K = 4 + 5l \qquad l \equiv 0,1,2,3,4 \mod 5$

$K \equiv 4, 9, 14, 19, 24 \mod 25.$

A similar argument shows

$K \equiv 24 \mod 25.$

$$\begin{cases} K \equiv 1 \mod 4 \\ K \equiv 24 \mod 25 \end{cases} \Longrightarrow \quad K \equiv 49 \mod 100$$

So $2^{49} \equiv 50 \mod 100$.

Point: Discrete log is

<u>HARD</u>.

How does this apply to cryptography?

Creating encryption keys!

Diffie-Hellman

Create a private key in a public communication channel.

- p large prime, g generator mod p. g and p are both public.

- Alice and Bob want to create a secret key that only they will know.

Alice: a $1 \le a \le p-2$ Private
Bob b: b $1 \le p \le p-2$ Keys

Alice computes $A \equiv g^a \bmod p$

Bob computes $B \equiv g^b \bmod p$

A and B are their public keys.

Alice and Bob can now both compute

$$A^b \equiv B^a \equiv g^{ab} \bmod p.$$

$Z = g^{ab} \bmod p$ is their shared private key.

An eavesdropper will only know p, g, g^a, g^b. Finding $g^{ab} \bmod p$ is the Discrete log problem, so is hard to crack.

Alice and Bob can now use their shared private key to set up encrypted communication with an appropriate method (e.g. AES).

<u>Ex:</u> $p = 101741$ $g = 7117$

Alice: $a = 52380$ $A = 10947$

Bob: $b = 66235$ $B = 56997$

$Z = 54692.$

this is their key to use for

encryption.