CS70: Lecture 11. Outline.

1. Public Key Cryptography
2. RSA system
   2.1 Efficiency: Repeated Squaring.
   2.2 Correctness: Fermat's Theorem.
   2.3 Construction.
3. Warnings.

Lots of Mods

$x = 5 \pmod 7$ and $x = 3 \pmod 5$.

What is $x \pmod{35}$?

Let's try 5. Not $3 \pmod 5$!
Let's try 3. Not $5 \pmod 7$!

If $x = 6 \pmod 7$
then x is in $\{5, 12, 19, 26, 33\}$.

Oh, only 33 is $3 \pmod 5$.

Hmmm... only one solution.

A bit slow for large values.

Simple Chinese Remainder Theorem.

My love is won. Zero and One. Nothing and nothing done.

Find $x = a \pmod m$ and $x = b \pmod n$ where gcd(m,n)=1.

CRT Thm: Unique solution $\pmod{mn}$.

Proof:
Consider $u = n(n^{-1} \pmod m)$.
$u = 0 \pmod n$ $u = 1 \pmod m$

Consider $v = m(m^{-1} \pmod n)$.
$v = 1 \pmod n$ $v = 0 \pmod m$

Let $x = au + bv$.
$x = a \pmod m$ since $bv = 0 \pmod m$ and $au = a \pmod m$
$x = b \pmod n$ since $au = 0 \pmod n$ and $bv = b \pmod n$

Only solution? If not, two solutions, x and y.
$(x - y) \equiv 0 \pmod m$ and $(x - y) \equiv 0 \pmod n$.
$\implies (x - y)$ is multiple of m and n since gcd(m,n)=1.
$\implies x - y \geq mn \implies x, y \notin \{0, \ldots, mn - 1\}$.
Thus, only one solution modulo mn. $\square$

Xor

Computer Science:
1 - True
0 - False

$1 \vee 1 = 1$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

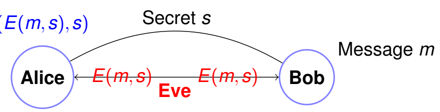
$A \oplus B$ - Exclusive or.
$1 \vee 1 = 0$
$1 \vee 0 = 1$
$0 \vee 1 = 1$
$0 \vee 0 = 0$

Note: Also modular addition modulo 2!
$\{0, 1\}$ is set. Take remainder for 2.

Property: $A \oplus B \oplus B = A$.
By cases: $1 \oplus 1 \oplus 1 = 1$. ...

Cryptography ...

$m = D(E(m,s), s)$



Example:
One-time Pad: secret s is string of length $|m|$.
$m = 10101011110101101$
$s =$ ....
$E(m,s)$ – bitwise $m \oplus s$.
$D(x,s)$ – bitwise $x \oplus s$.
Works because $m \oplus s \oplus s = m$!
...and totally secure!
...given $E(m,s)$ any message m is equally likely.

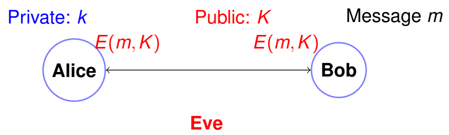
Disadvantages:

Shared secret!

Uses up one time pad..or less and less secure.

Public key crypography.

$m = D(E(m,K), k)$



Everyone knows key K!
Bob (and Eve and me and you and you ...) can encode.
Only Alice knows the secret key k for public key K.
(Only?) Alice can decode with k.

Is this even possible?

Is public key crypto possible?

We don't really know.
...but we do it every day!!!

RSA (Rivest, Shamir, and Adleman)
Pick two large primes p and q. Let $N = pq$.
Choose e relatively prime to $(p-1)(q-1)$.[1]
Compute $d = e^{-1} \mod (p-1)(q-1)$.
Announce $N (= p \cdot q)$ and e: $K = (N, e)$ is my public key!

Encoding: $\mod (x^e, N)$.

Decoding: $\mod (y^d, N)$.

Does $D(E(m)) = m^{ed} = m \mod N$?

Yes!

[1] Typically small, say $e = 3$.

Iterative Extended GCD.

Example: $p = 7$, $q = 11$.

$N = 77$.
$(p-1)(q-1) = 60$
Choose $e = 7$, since $\gcd(7, 60) = 1$.
egcd(7,60).

$$\begin{aligned} 7(0) + 60(1) &= 60 \\ 7(1) + 60(0) &= 7 \\ 7(-8) + 60(1) &= 4 \\ 7(9) + 60(-1) &= 3 \\ 7(-17) + 60(2) &= 1 \end{aligned}$$

Confirm: $-119 + 120 = 1$

$d = e^{-1} = -17 = 43 = \pmod{60}$

Encryption/Decryption Techniques.

Public Key: $(77, 7)$
Message Choices: $\{0, \ldots, 76\}$.

Message: 2!

$E(2) = 2^e = 2^7 \equiv 128 \pmod{77} = 51 \pmod{77}$
$D(51) = 51^{43} \pmod{77}$
uh oh!

Obvious way: 43 multiplications. Ouch.

In general, $O(N)$ or $O(2^n)$ multiplications!

Repeated squaring.

Notice: $43 = 32 + 8 + 2 + 1$. $51^{43} = 51^{32+8+2+1} = 51^{32} \cdot 51^8 \cdot 51^2 \cdot 51^1 \pmod{77}$.
4 multiplications sort of...
Need to compute $51^{32} \ldots 51^1$.?
$51^1 \equiv 51 \pmod{77}$
$51^2 = (51) * (51) = 2601 \equiv 60 \pmod{77}$
$51^4 = (51^2) * (51^2) = 60 * 60 = 3600 \equiv 58 \pmod{77}$
$51^8 = (51^4) * (51^4) = 58 * 58 = 3364 \equiv 53 \pmod{77}$
$51^{16} = (51^8) * (51^8) = 53 * 53 = 2809 \equiv 37 \pmod{77}$
$51^{32} = (51^{16}) * (51^{16}) = 37 * 37 = 1369 \equiv 60 \pmod{77}$

5 more multiplications.

$51^{32} \cdot 51^8 \cdot 51^2 \cdot 51^1 = (60) * (53) * (60) * (51) \equiv 2 \pmod{77}$.

Decoding got the message back!

Repeated Squaring took 9 multiplications versus 43.

Repeated Squaring: x^y

Repeated squaring $O(\log y)$ multiplications versus y!!!

1. x^y: Compute $x^1, x^2, x^4, \ldots, x^{2^{\lfloor \log y \rfloor}}$.
2. Multiply together x^i where the $(\log(i))$th bit of y (in binary) is 1.
   Example: $43 = 101011$ in binary.
   $x^{43} = x^{32} * x^8 * x^2 * x^1$.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers. Repeated Squaring:
$O(n)$ multiplications.
$O(n^2)$ time per multiplication.
$\Longrightarrow O(n^3)$ time.
Conclusion: $x^y \mod N$ takes $O(n^3)$ time.

RSA is pretty fast.

Modular Exponentiation: $x^y \mod N$. All n-bit numbers.
$O(n^3)$ time.

Remember RSA encoding/decoding!

$E(m, (N, e)) = m^e \pmod{N}$.
$D(m, (N, d)) = m^d \pmod{N}$.

For 512 bits, a few hundred million operations.
Easy, peasey.

Decoding.

$E(m,(N,e)) = m^e \pmod N$.
$D(m,(N,d)) = m^d \pmod N$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod N$.

Always decode correctly?

$E(m,(N,e)) = m^e \pmod N$.
$D(m,(N,d)) = m^d \pmod N$.

$N = pq$ and $d = e^{-1} \pmod{(p-1)(q-1)}$.

Want: $(m^e)^d = m^{ed} = m \pmod N$.

Another view:
$d = e^{-1} \pmod{(p-1)(q-1)} \iff ed = k(p-1)(q-1)+1$.

Consider...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod p$,

$a^{p-1} \equiv 1 \pmod p$.

$\implies a^{k(p-1)} \equiv 1 \pmod p \implies a^{k(p-1)+1} = a \pmod p$

versus $a^{k(p-1)(q-1)+1} = a \pmod{pq}$.

Similar, not same, but useful.

Correct decoding...

Fermat's Little Theorem: For prime p, and $a \not\equiv 0 \pmod p$,

$a^{p-1} \equiv 1 \pmod p$.

Proof: Consider $S = \{a \cdot 1, \ldots, a \cdot (p-1)\}$.

All different modulo p since a has an inverse modulo p.
S contains representative of $\{1, \ldots, p-1\}$ modulo p.

$$(a \cdot 1) \cdot (a \cdot 2) \cdots (a \cdot (p-1)) \equiv 1 \cdot 2 \cdots (p-1) \mod p,$$

Since multiplication is commutative.

$$a^{(p-1)}(1 \cdots (p-1)) \equiv (1 \cdots (p-1)) \mod p.$$

Each of $2, \ldots (p-1)$ has an inverse modulo p, solve to get...

$$a^{(p-1)} \equiv 1 \mod p.$$

$\square$