# Class Notes, November $8^{th}$

Dalton Watts

Recall RSA: Public Key Cryptography
Basic Steps:
Pick 2 big prime numbers, p and q
$n = p \times q$
Pick e where $GCD(e, \varphi(n)) = 1$
Secretly compute $d \equiv e^{-1} \pmod{\varphi(n)}$
Encryption: $E_{n,e}(m) = m^e \pmod{n}$ which anyone can do.
Decryption: $D_{n,d}(c) = c^d \pmod{n}$ which only Bob can do
To break RSA, Eve needs to factor n (which is very hard to do).

Timing attacks (discovered by an undergrad in 1994)

If you can time how long it takes Bob to do decryption you can use this to figure out what d is.

Now systems add a small delay in the decryption Step to protect against this.

More info on Timing Attacks in the book.

We'll be focusing more on the direct mathematical attack. What does it take to do it?

By default, we know m, n, e, and c. We don't know p, q, nor d.

In order to implement RSA Bob needs to find 2 large primes p and q. He does this by following three steps:

1) Chooses a random 120 digit number

2) Check to see if this number is prime

3) If it is, use it. If it isn't, repeat from step 1.

How many primes are there? Infinitely many.

Less than x, though? Define the prime counting function $\pi(x)$ = count how many primes are less than or equal to x.

$\pi(10) = 4$

$\pi(10.5) = 4$

$\pi(11) = 5$

$\pi(1987) = 300$

Number of 120 digit primes would be $\pi(10^{121}) - \pi(10^{120})$

$\pi(x) \approx \frac{x}{\ln(x)}$

1900 Mathematicians proved that this is a good approximation.

Prime number theorem: $\pi(x) = \frac{x}{\ln(x)} + O(\frac{x}{(ln(x))^2})$

Better yet, some Calculus! $\pi(x) = \int_2^x (\frac{1}{\ln(t)})dt + O\left(\frac{x}{e^{\sqrt{\ln x}}}\right)$

Estimate for the number of 120 digit primes is approximately $3.2 * 10^{118}$

Probability of choosing a 120 digit number that's prime is $3.2*10^{118}/(10^{121} - 10^{120})$ which is approximately $\frac{1}{260}$.

If we ignore even numbers (because all of them are divisible by 2), the probability increases to $\frac{1}{130}$.

We can include parameters for other low primes, to increase the probability even more, but even at 130 random picks, it isn't hard for a computer to find a prime number.

Now, how do we implement step 2?

Method 1: Check to see if the number is divisible by anything less than its square root. But, if n has 120 digits, then the square root of n has about 60 digits. That's still pretty big. Even if you could divide a trillion numbers every second, this still takes longer than the age of the universe.

Previously discussed is the Fermat "Primality" test.

Check if n is prime by picking a variable $a$ such that $1 < a < n - 1$

Compute $a^{n-1} \pmod{n}$

If this is not 1, return "composite"

Otherwise, it's probably prime.

Repeat for several values of a to be more sure.

It's not proven to be prime with this method, but it's at least pseudo-prime. Unfortunately that's not good enough, and There also exist Carmichael numbers that act as pseudo-primes for every base $a$.

Method 2: Miller-Rabin Test

Like Fermat, it is probabilistic. It's probably a prime, and it's less likely that you'll get pseudo-primes with this method, although it is still possible.

Check to see if $n$ is prime.

For some integer $k$ and odd integer $m$, write $n-1 = 2^k m$.

Pick a random variable $a$ between $1 < a < n-1$.

Step 1: Compute $b_0 \equiv a^m \pmod{n}$.

If $b_0 \equiv \pm 1 \pmod{n}$ return "Probably Prime"

Else: For i in $[1..k-1]$:

Compute $b_i \equiv b_{i-1}^2 \pmod{n}$.

If $b_i \equiv 1 \pmod{n}$ return "Composite!"

If $b_i \equiv -1 \pmod{n}$ return "Probably Prime!"

If $b_{k-1} \neq \pm 1$ return "Composite!"

If we want to be more certain after getting "Probably Prime" Try again with another a.

Example 1: $n = 13, a = 2$

Write $13 - 1 = 2^2 * 3$ such that $k = 2, m = 3$

Compute $b_0 \equiv 2^3 \pmod{13}$

$b_0 \equiv 8 \pmod{13}$

$b_1 \equiv b_0^2 \equiv 8^2 \equiv 64 \equiv -1 \pmod{13}$

Example 2: $n = 121, a = 3$

$n - 1 = 2^3 * 15$

$b_0 = a^{15} \equiv 3^{15} \equiv 1 \pmod{121}$

Miller-Rabin says 121 is "Probably Prime"

We call this a strong-pseudo prime. There are no "Carmichael-like" numbers for Miller-Rabin.

If n is any composite number then at least $\frac{3}{4}$ of the choices for $a$ prove that n is composite usually this is much higher.

If you run Miller-Rabin 10 times with different values for $a$, if it's still probably prime, the probability that it's actually composite is less than $(\frac{1}{4})^{10}$ which is less than one-in-a-million. If you run it 20 times, you get less than (one-in-a-million)$^2$

Miller-Rabin is still probabilistic

You can make it deterministic (provable prime) if you have to assume the Riemann hypothesis is true. Unfortunately, nobody has proven it, and there's still a mathematical bounty available for anyone that can prove it.

In 2003, it was proved that proving a number is prime is a polynomial running time problem.

The algorithm "AKS" is too slow to be useful.

How would we attack RSA? Factoring, how do we do Factoring quickly?

What is the fastest way to factor?

Method 1: Want to factor n

Try dividing $n$ by $2, 3, 4 \ldots$ continue until you find a factor of $n$. (Note: you can stop when you get to the square root.) This can take up to $\sqrt{n}$ steps, which is way to slow for $n$ the size used in RSA.

Basic Principle of Factoring: If you can find 2 numbers $x, y$

where $x^2 \equiv y^2 \pmod{n}$ but $x \not\equiv \pm y \pmod{n}$

Then $n$ is composite and $\gcd(x - y, n)$ gives a nontrivial factor of $n$. (not 1 or $n$)

Proof: Suppose those two things are true. Let $d = \gcd(x - y, n)$

Case 1: $d = n$. Then $n$ divides $x - y$.

So, $x - y \equiv 0 \pmod{n}$

Such that $x \equiv y \pmod{n}$

We assume this doesn't happen.

Case 2:$d = 1$

Since $x^2 \equiv y^2 \pmod{n}$

$x^2 - y^2 \equiv 0 \pmod{n}$

So n divides $x^2 - y^2 = (x - y)(x + y)$

Since $\gcd(x - y, n) = 1$

So, $n$ divides $x + y$

then $x + y \equiv 0 \pmod{n}$

$x \equiv -y \pmod{n}$

Which we assumed was not the case. Thus $d$ is a factor of $n$ which is neither 1 nor $n$.