MATH 314 Fall 2023 - Class Notes

4/1/2024

Scribe: Andrew Mandahl

Summary: On this day we learned about the Modular Exponentiation technique, which allows us to calculate $a^b \pmod m$, since the numbers can never get larger than m^2. We also learned about the 3 pass protocol which is a technique of public key encryption, this technique is however rarely used due to the inefficiency of the technique caused by necessitating 3 separate transmissions which utilizes a large amount of bandwidth.

Notes:

Modular Exponentiation: This lets us compute $a^b \pmod m$, this technique is very quick, numbers involved never get larger than m^2, If the modulus is prime then we can do even quicker computations, this is Fermat's "little" Theorem. It states that If p is prime and a is not 0(mod p), then $a^{p-1}=1 \pmod p$

Ex: P = 5, A = 3

$3^4 \pmod 5$

$3^2 = 9 = 4 \pmod 5$

$3^4 = 4^2 = 16 = 1 \pmod 5$

Ex 2: P = 11, A=2

$2^{10} = \pmod{10}$

$2^2 = 4 \pmod{11}$

$2^4 = 4^2 = 16 = 5 \pmod{11}$

$2^8 = 5^2 = 25 = 3 \pmod{11}$

$2^{10} = 2^8 + 2^2 = 3(4) = 12 = 1 \pmod{11}$

"Proof of Fermat's little theorem" == FLT

Since a is not 0(mod p)

Take all numbers between 1 and p-1, then multiply all of them by a:

1*a, 2*a, 3*a, … (p-1)a (mod p)

a has an inverse and all the list for multiplication (1*a, 2*a, 3*a, … (p-1)a (mod p)) are different (mod p), which means numbers 1,2, …, p-1 are all in the list just in some different order.

Multiply these all together

(1*a)(2*a)(3*a)...((p-1)(a)) (mod p) == 1 * 2 *3 *, ... * (p-1) (mod p)

All the numbers 1,2, ... p-1 have inverses and thus can be used to cancel numbers out, therefore this means that $a^{p-1}=1$ (mod p)

How can we use Fermat's theorem:

Lets say we want to solve 3^{23} (mod 11)

FLT tells us that $3^{10} = 1$(mod 11)

Therefore $3^{20} – (3^{10})^2 = 1$(mod 11)

Therefore 3^{23}(mod 11) == 3^3 (mod 11)

Therefore we only need to solve for $3^{23} = 3^{20} * 3^3$(mod 11) == $1 * 3^3$(mod 11)

3^3 == 27 == 5 (mod 11) Therefore 3^{23} == 5(mod 11)

Ex: Find 7^{61}(mod 31)

7^{61}== $7^{32+16+8+4+1}$

FTL states: 31 is prime so $7^{30} = 1$ (mod 31)

$7^{61} = 7^{60} * 7^1$ == 1(7) = 7(mod 31) = $(7^{30})^2 * 7^1$

Basic principle of exponents modulo a prime p If your equation is mod p), treat all exponents (mod p-1)

Ex: 5^{50} (mod 19)

Work mod (18) in the exponent, so 50 = 14 (mod 18), 50 == 5^{14}(mod18)

We can utilize this tequnique to "undo" something to an exponent E(x) = E(x) = x^a (mod p)

Ex: E(x) – x^5(mod 17), So lets use it to decrypt this example,

Guesss (D(y) == y^d(mod p)

D(E(x)) = x == $(x^5)^d$ == x^1 (mod 17)

$X^{3d} = x^1 = x^1$(mod 17))

Need

$5d == 1 (mod 16)$

Find 5^{-1} (mod 1C)

Euclid: 16/5 = 3 R(1)

16 = 3(5) +1

1= 16 – 3(5)

$5^{-1} == -3 == 13 (mod\ 16)$

$E(x) == x^5 (mod\ 17)$

$D(y) == y^{13} (mod\ 17)$

$(X^5)^{13} == x^{65} == x^{64} * x^1 == (x^{16})^4 * x^1) (mod\ 17) == x (mod\ 17)$, Is $tE(x) = x^e (mod\ p)$ better than $E(x) = ax+b (mod\ p)$?, If you know $a^e == b (mod\ p)$, you know a and b but not e.

Solving for e is difficult which is called the discrete log problem, If this wasn't mod p then we can solve for e using basic logarithm rules => $a^e = b$ solve for e, $log(a^2) = log(b)$

$Elog(a) = log(b)$, therefore $e= log(b)/log(a)$, Note that this never works in mod p.

3 pass protocol:

Physical world vrsn

Alice has a box she locks it using her padlock and she mails the box to Bob, then Bob add his own padlocks, then bob sends it back to Alice who then unlocks her lock and resends the package to Bob. Then Bob unlocks his lock using his key and then can retrieve whatever was in the box

3 pass protocol mathematical version: Alice picks a large prime P, and tells everyone the P value, she then picks an encryption exponent a where the gcd(a,p-1) = 1

Then Alice can compute the dectryption exponent $a^{-1} (mod\ p-1) == A$, then Bob who also know the P value, pica a b with gcd(b,p-1) = 1 then computes $B = b^{-1} (mod\ p-1)$

Meaning we now have

$E_A = x^a (mod\ p)$

$D_A(y) = y^A (mod\ p)$

$E_B = x^b (mod\ p)$

$D_B(y) = y^B (mod\ p)$,

Then Alice wants to send the plaintext, m(mod p), she computes $C_1 = E_a(m) = m^a (mod\ p)$ then sends that to Bob.

Then Bob encrypts it again: $C_2 = E_B(c_1) = C_1^b (mod\ p)$ and sends that back to Alice

Then Alice decrypts, $C_3 == D_A(C_2) = C_2^A \pmod p$, then sends that to Bob.

Then Bob decrypts $D(C_3) = C_3^B \pmod p$, and then recovers m, which results in there never being a key or an unencrypted file being sent.