

9/27 0:26:00

Public Key Cryptography

Bob

- Selects 2 large primes p and q and mult. them to get n . He has K_{public} and K_{private} . These 2 keys can be the $m1$ of each other.
- Sends his public key to Alice.

Alice - creates a master key K and uses Bob's public key to enc. K .

- She sends the encrypted master key to Bob.

Bob

- Uses his private key to dec. Alice's encrypted master key. He now has Alice's master key K .
- This is called a key agreement.
- Now they both have the master key so they can use 3DES, AES, etc.

One Way Functions



3 well known one-way functions

① Factoring integers (used in RSA)

- given a composite (not prime) integer, find its prime factors. e.g. given 2 primes p and q , you can easily calculate $n = pq$. But given n , it's computationally infeasible to factor n into 2 primes if p and q are large.

② Discrete Log (used in DH, ElGamal, D/E SA)

given a , y , and mod m , find x s.t.

$$a^x = y \pmod{m}$$

or write like this:

$$\log_a y = x \pmod{m}$$

read like this: $y \xleftarrow{a} x \quad a^x = y$

③ Elliptic Curve (EC)

- a generalization of ②
- out of scope for this class

New Math concepts

Euler's phi function 9/27 0:40:00

given a set of m integers $\{0, 1, \dots, m-1\}$
how many numbers in the set are relatively
prime to m ?

e.g. set = $\{0, 1, 2, 3, 4, 5\}$ and $m=6$
how many are relatively prime to 6?

$$\gcd(0, 6) = 6$$

$$\gcd(1, 6) = 1$$

$$\gcd(2, 6) = 2$$

$$\gcd(3, 6) = 3$$

$$\gcd(4, 6) = 2$$

$$\gcd(5, 6) = 1$$

relatively prime means

$$\gcd(a, b) = 1$$

$\gcd$ is 1, so we have
2 that are relatively
prime to 6.

5 and 1 are relatively prime / coprime to 6

Therefore, the phi function of 6

$$\phi(6) = 2$$

e.g. 2: $m = 5$ so $\{0, 1, 2, 3, 4\}$

$$\gcd(0, 5) = 5$$

$$\gcd(1, 5) = 1$$

$$\gcd(2, 5) = 1$$

$$\gcd(3, 5) = 1$$

$$\gcd(4, 5) = 1$$

$$\text{so } \phi(5) = 4$$

9/27 0:44:00 A better way to calculate ϕ

The prime factorization of m

$$m = p_1^{e_1} \cdot p_2^{e_2} \cdots p_n^{e_n}$$

then

$$\phi(m) = \prod_{i=1}^n (p_i^{e_i} - p_i^{e_i-1})$$

With this equation, we can calculate the phi of any value through the prime factorization.
e.g.

$$m = 6 = \underset{\substack{\uparrow \\ p_1}}{2} \cdot \underset{\substack{\uparrow \\ p_2}}{3} \quad \begin{matrix} \nwarrow e_1 \\ \nearrow e_2 \end{matrix}$$

$$p_1 = 2 \quad e_1 = 1$$

$$p_2 = 3 \quad e_2 = 1$$

$$\phi(6) = \underbrace{(p_1^{e_1} - p_1^{e_1-1})}_{i=1} \cdot \underbrace{(p_2^{e_2} - p_2^{e_2-1})}_{i=2}$$

$$= \underbrace{(2^1 - 2^{(1-1=0)})}_{i=1} \cdot (3^1 - 3^0) = (2-1)(3-1) = 1 \cdot 2 = 2$$

e.g. find $\phi(5)$

1. Find prime factorization of 5.

$$5 = 5^1 \cdot 1^1 \quad p_1 = 5 \quad e_1 = 1 \quad n = 1$$

$$(5^1 - 1) = 5 - 1 = 4$$

Also interesting, if m is the product of 2 prime no., $m = \underset{p}{\text{prime}_1} \cdot \underset{q}{\text{prime}_2}$.

$$\text{then } \phi(m) = (p-1)(q-1)$$

Euler's theorem: generalization of Fermat's Little thm. ($a^{p-1} \equiv 1 \pmod{p}$)

$$a^{\phi(m)} \equiv 1 \pmod{m}$$

note: $\phi(m) = p-1$

e.g. $m = 12 \quad a = 5$

$$\gcd(12, 5) = 1$$

do prime factorization
for 12: $12 = 2^2 \times 3^1 = e_1 \cdot e_2$
 $p_1 \quad n=2 \quad p_2$

$$\phi(12) = (2^2 - 2^1)(3^1 - 3^0) = 4$$

Verify

$$5^{\phi(12)} \equiv 1 \pmod{12}$$

$$5^4 \equiv 625 \equiv 1 \pmod{\text{red. over } 12}$$

1976 - Researchers defined the problem of public key enc.

K_{public} - for enc.

K_{private} - for dec.

1977 - Ronald Rivest, Adi Shamir, Leonard Adleman provided solution: RSA

Key generation of RSA (first part)

- Done by the receiver/person who wants to receive the secret msg.

① Generate 2 large random primes of similar/same size: p, q .

② Compute $n = pq$ and $\phi = (p-1)(q-1)$
fyi: $(p-1)(q-1)$ shows how many numbers in this set are coprime to n .

$e = \text{public key}$ ③ Select a random int. e (the public key) $\rightarrow$
104:38 $(1 < e < \phi)$ s.t. $\gcd(e, \phi) = 1$. Use Euc. Alg. to verify. If $\gcd \neq 1$, generate a new no.

$d = \text{private key}$ ④ Use the Ext. Euc. Alg. to compute another unique int. (the private key) d where $(1 < d < \phi)$ s.t.
 $e \cdot d \equiv 1 \pmod{\phi}$

⑤ Transmit (n, e) as public values and the private key d remains secret.

to make it hard to find the
one with one

one not - string

one not - string

1) 1000, 2) 1000, 3) 1000 - 1000
1000 - 1000

(1000) [1000 to 1000]

show the result of the
to see the result of

1) Generate a large random number
2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

(1-p)(1-q) = 1 - pq = 1 - 1000

1000 - 1000 = 1000 - 1000 = 1000

③ Randomly generate e , check to see if it's coprime w. respect to ϕ then in 2000
④, you use ext-euc. alg. and find its m .

1) 1000, 2) 1000, 3) 1000 - 1000
1000 - 1000 = 1000 - 1000 = 1000

1000 - 1000 = 1000 - 1000 = 1000

RSA

Alice
Enc.

1. obtain n and e .
2. msg space will be between 0 and $n-1$
3. Compute cipher text as follows $C = m^e \pmod{n}$ *
 - So everytime you pick up a part of your msg / plaintext, convert it to a number and multiply that to e (public key) and do the modular reduction over n . Result is cipher text.
4. Send C to other party

Bob
Dec.

1. $m = C^d \pmod{n}$
- retrieve msg by calculating this

* Use modular exponentiation algorithm.

10/2

Algorithms needed for RSA

- ① Random Number Generator - to produce 2 large numbers.
- ② Primality Test - checks if the random no. we generated are prime. There are 2 of these.
- ③ Euclidian - used to compute gcd. GCD has to be = 1 aka coprime condition aka relatively prime. We will use this to see if "e" is a suitable public key - because we need to find the inverse of "e" / public key which is the private key.
- ④ Extended Euclidian - Used to find the inverse (m1) of "e", which we call "d".
- ⑤ Modular Exponentiation - To do enc. / dec.

enc. $C = m^e \pmod{n}$ Alice

Diagram: A green line connects 'C' to 'ciphertext' below it. A red line connects 'e' to 'public key' above it. A red line connects 'm' to 'plaintext' below it.

dec. $m = C^d \pmod{n}$ Bob

Diagram: A red line connects 'm' to 'plaintext' below it. A green line connects 'C' to 'ciphertext' below it. A red line connects 'd' to 'private key' above it.

10/2 0:15:00

RSA Example

Alice

- wants to send the plaintext 4. So $x = 4$.

Bob

- generates these numbers:

- $p = 3, q = 11$

- $n = pq = 3 \cdot 11 = 33$

- $\phi(n) = (p-1)(q-1) = 20$

- $e = 3$

- $d = \text{inverse of } e, d = e^{-1} \equiv 7 \pmod{20}$

- p and q should be 2048 bits as of today, was 256 10 yrs ago.
- $\phi(n) = 20$, 20 numbers in this set will be coprime w. respect to n . This means that if you randomly select a no. " e ", then there is a high chance that " e " is invertible - that you can find its $m1$. One condition of RSA was that the $\gcd$ of " e ", - the public key - w respect to ϕ must be 1.
- e is randomly chosen. It is recommended to be as large as the primes, but not necessary. It just needs to satisfy $\gcd(e, \phi) = 1$. Verify $\gcd(3, 20) = 1$.
- d is inverse of e . Verify: $3 \cdot 7 \stackrel{20}{\equiv} 1, 3 \cdot 7 \pmod{20} = 1$. So e and d are $m1$ of each other.

Send public key to Alice $K_{\text{public}}(\overset{n}{33}, \overset{e}{3})$

- Alice will calculate

$$y = 4^3 \pmod{33} = 31$$

↑ ↑ ↖
 plaintext to n cipher text
 the power of "e"

she sends 31 to Bob

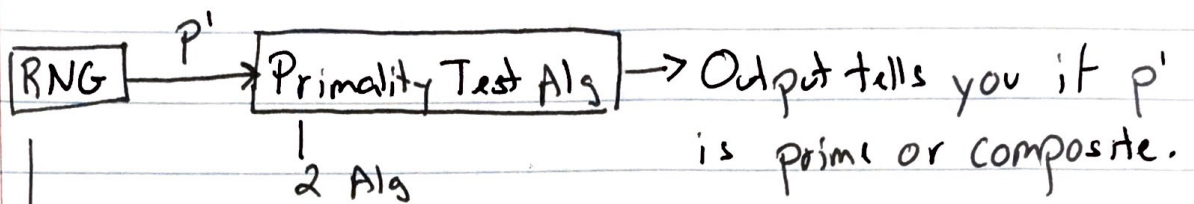
- Bob will calculate:

$31^7 \equiv 4 \pmod{33}$
 ↑ ↖
 ciphertext plaintext from Alice

⊕ Simple math for enc. dec.

② if the numbers are large integers, then even using fast alg. like modular exponentiation, it's very time consuming to do enc./dec.

10/2 0:23:00



Use random no. generators to generate a potential prime.

There are 2 Primality Test Alg. They are probabilistic - Tells you w. a high probability if a no. is prime, but not 100%.

① Fermat's Primality test

② Miller - Rabin Primality test

- The parameters to the Alg will define the accuracy of the results.

Modular Exponentiation examples

Repeated Square + Multiply alg. for exponentiation this finite field $\mathbb{Z}_n$ pronounced $\mathbb{Z} \bmod n$. $\mathbb{Z}_n$ is a set of int limited to n .

Input: an int "a" in our finite field.
an int "k".

$a \in \mathbb{Z}_n$, $0 \leq k < n$ $\xrightarrow{\text{calculate}}$ binary

representation of k :

$t = \text{total bits} / k = \text{power?}$

$$k = \sum_{i=0}^t k_i 2^i$$

k_i can be 0 or 1, even or odd.

Output: $a^k \pmod n$

- ① set $b \leftarrow 1$, if $k = 0$, return b . x^0 is 1, so return 1 if $k=0$
- ② set $A \leftarrow a$
- ③ if $k_0 = 1$, set $b \leftarrow a \parallel (01)_2$. $k=1$ means no. is odd.
- ④ Loop $i = 1 \dots t$
 - ④.1 $A \leftarrow A^2 \pmod n$
 - ④.2 if $k_i = 1$, $b \leftarrow A \cdot b \pmod n$
- ⑤ Return b

e.g. Calculate a^{26} using repeated square and multiply alg.

$K = 26$, don't care about a .

First, find the binary representation of 26.

Base Conversion:

$$26 = 2 \times 13 + 0$$

$$13 = 2 \times 6 + 1$$

$$6 = 2 \times 3 + 0$$

$$3 = 2 \times 1 + 1$$

$$1 = 2 \times 0 + 1$$

Divide by 2 repeatedly until $q == 0$.

$$r = b \cdot q + a$$

Read result up $\uparrow$

$$26 = 1 \ 1 \ 0 \ 1 \ 0$$

① $b \leftarrow 1$. $K == 0$? No, don't return! ~~K_4~~ $K_4 K_3 K_2 K_1 K_0$

② $A \leftarrow a$

③ if $K_0 == 1$ (odd) set $b \leftarrow a$.

④ Loop $i = 1 \dots 4$ // loop through all digits.

10/2
0:35:00

$$\boxed{i=1} \quad A \leftarrow A^2 \pmod n$$

$$K_1 == 1? \text{ Yes. } b \leftarrow A \cdot b \pmod n$$

$$\leftarrow A^2 \cdot b$$

$$\boxed{i=2} \quad A \leftarrow (A^2)^2$$

$$\leftarrow A^2 \cdot 1 \pmod n \text{ Result: } A^2$$

$$A \leftarrow A^4 \pmod n$$

$$K_2 == 1? \text{ No}$$

$$\boxed{i=3}$$

$$A = A^8 \pmod n, K_3 \text{ is } 1 \text{ so } b \leftarrow A^8 \cdot A^2 = A^{10} \pmod n$$

$$\boxed{i=4} \quad A = (A^8)^2 = A^{16} \pmod n, K_4 \text{ is } 1 \text{ so } b \leftarrow A^{16} \cdot A^{10} = A^{26} \pmod n$$

⑤ Return $A^{26} \pmod n$ // Return b .

e.g. Calculate a^7

$$7 = (111)_2$$

$$\begin{array}{c} 111 \\ / \quad \backslash \end{array}$$

$$K_2 \ K_1 \ K_0 \quad t=2$$

① $b \leftarrow 1$

$K == 0$? No.

② $A \leftarrow a$

③ Check if no. is even or odd. $K_0 = 1$, odd
so $b \leftarrow a$

④ Loop 1..2

$$\boxed{i=1}$$

$$A = A^2 \pmod n$$

$$\text{if } K_1 == 1, \text{ then } b = A^2 \cdot b = A^3 \pmod n$$

$$\boxed{i=2}$$

$$A = (A^2)^2 = A^4 \pmod n$$

if $K_2 == 1$, then $b = \text{latest value of } A$, which is $A^4 \times \text{latest value of } b$
which is $A^3 \pmod n = A^7 \pmod n$

⑤ Return $A^7 \pmod n$

Primality Tests

① Fermat's primality Test Alg.

Input: An odd int $n \geq 3$ (even no. cant be prime)
A security parameter $t \geq 1$. This is the no. of iterations. The more tests, the more accurate.

Output: either "prime" or "composite"

① For $i = 1 \dots t$

Chose a random int $2 \leq a \leq n-2$. Why

①.1 $n-2$? cuz $n-1$ could be even.

①.2 Compute $r = a^{n-1} \pmod n$ using square+multiply alg.

①.3 if $r \neq 1$, we found a counter example (composite) return it.
- if after t loops we don't find any composites, return "it's prime".

We can set $t = 2000$ and try 2000 times to find a counter example. Why is this the case? Back to Fermat's Little thm: any int to the power of $p-1 \equiv 1 \pmod p$ when p is prime. If $r = a^{n-1}$ is equal to 1, then n is probabably prime, but we try many times. If we get anything other than 1, that is a counter example.

- Find online a good size of t if you want to generate a prime of size 512, 1000, etc.

e.g. $n = 25$ (not prime example)
 $t = 3$

$i = 1, \dots, 3$

$[i = 1]$ generate a random no. $2 \leq a = 9 \leq n-2$ ²³
compute $r = a^{n-1} \pmod{n}$

$$r = 9^{24} \pmod{25} = 11$$

$r \neq 1$, return "composite"

e.g. $n = 25$
 $t = 3$

$[i = 1]$ $a = 7$

$$r = 7^{24} \pmod{25} = 1 \text{ // round 1 OK}$$

$[i = 2]$ $a = 4$

$$r = 4^{24} \pmod{25} = 6$$

$r \neq 1$, return "composite"

e.g.

$n = 25$

$t = 2$

$[i = 1]$ $a = 7$

$$r = 1, \text{ OK}$$

$[i = 2]$ $a = 18$

$$r = 1, \text{ OK}$$

Return "prime" - mistake, 25 is not prime.

10/2 1:04 Miller Primality Test

security parameter

Input: an odd no. $n \geq 3$, $t \geq 1$
Output: "prime" or "not prime"

① we write $n-1$ as $2^s \cdot r$ s.t. r is odd

2 to any power is even $\xrightarrow{\text{even}} \underbrace{\text{even} \cdot \text{odd}}_{\text{even}}$ so therefore $n-1$ is even

e.g. if $n=25$

then $n-1=24$ and we write

24 as $2^3 \times 3$
 / r
 constant

to get $2^3 \cdot 3$, we did prime factorization on 24

② Loop $i=1..t$

②.1 chose a random int a . $2 \leq a \leq n-2$

②.2 compute y . $y = a^r \pmod{n}$ using the square + multiply algorithm. - always used v. modular exponentiation.

②.3



(2.3) if $y \neq 1$ and $y \neq n-1$

Loop:

$j = 1$

while $j \leq S-1$ and $y \neq n-1$

{ calculate y . $y \leftarrow y^2 \pmod n$
if $y = 1$, then return "composite"
 $j++$

if $y \neq n-1$, return "composite"
else

 return "prime"

ElGamal - Public Key Enc. Scheme

- Each person/entity creates the public key and corresponding private key
- Generate one (instead of 2 as in RSA) primes. This saves a little time.
- Generate alpha α of the multiplicative group $\mathbb{Z}_p^*$ of the integers mod p .

Note: $\mathbb{Z}_p = \{0, 1, \dots, p-1\}$ $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$

Note: You can select a random no.,
say 512 bits and then test it w. a
primality test alg. (for p , not α)

- Select a random int. " a " $1 \leq a \leq p-2$
then compute $\alpha^a \pmod{p}$ using sq+mult.

then the public key = (p, α, α^a)

" a " = private key

a single value

Algorithm for enc./dec. -

Message: m (p, α, α^a) " a "
 Public Private

Enc. -

- generated by receiver
- Obtain Alice's public key. Note: we need to add authenticity & integrity.
 - Represent the msg as an int or int sequence. $\{0, 1, \dots, p-1\}$
 - Select a random int k . $1 \leq k \leq p-2$
 - Compute gamma $\gamma = \alpha^k \pmod{p}$
using sq+mult.

Compute delta $\delta = m \cdot (\alpha^a)^k \pmod{p}$
 ^{msg} ^{single value}

- Ciphertext C has 2 parts γ and δ
- ⊖ ciphertext is larger than plaintext.
- ⊖ 2 rounds of calculations vs RSA (1).
RSA had 1 modular exponentiation, this has 2.

$$C = (\gamma, \delta)$$

- ⊕ Only need 1 prime no - can use an existing one so we save some time during key gen.

Dec.

- Costs less than RSA.
- Enc. costs more.

a) Receiver will use the private key "a" to compute:

$$\xrightarrow{\text{part of ciphertext}} Y^{(p-1-a)} \pmod{p}$$

Public Private key

calculating Y^{-a} is the main purpose here.

b) Recover m as follows:

$$m = Y^{-a} \cdot f \pmod{p}$$

$$Y^{p-1-a} \pmod{p} \equiv \underbrace{Y^{p-1}}_{=1} \cdot Y^{-a} \pmod{p}$$

$$m = Y^{-a} \pmod{p} \quad // \text{ we calculate } Y^{p-1-a} \text{ to get this}$$

key gen phase enc. phase

$$\text{Then we know } f = m \cdot (\alpha^a)^k \pmod{p} = m \cdot (\alpha^k)^a = \underline{m \cdot Y^a}$$

$$\alpha^k \text{ is } Y, \text{ so } f = m \cdot (Y)^a \text{ then } Y^{-a} \cdot f \pmod{p} = \underline{m \cdot Y^a \cdot Y^{-a} = m}$$

$$\log_x x = a \pmod{p} \rightarrow \alpha^a = x \pmod{p}$$

That's why ElGamal relies on Dh problem.

α, x, p are known. a is impossible to find.

e.g. ElGamal Enc.

$$\begin{array}{ccc} p = 2357 & \overset{\text{random}}{a} = 1751 & m = 2035 \\ \alpha = 2 & k = 1520 & \alpha^a \end{array}$$

$$Y = \alpha^k = 2^{1520} \pmod{2357} = 1430$$

$$f = m \cdot (\alpha^a)^k \pmod{p} = 2035 \cdot 1185^{1520} \pmod{2357} \\ = 697$$

$$C = (Y, f) = (1430, 697)$$

⊖ cipher text is double

⊖ enc. phase is more \$

Dec. - Calculate Y^{p-1-a} . The main purpose is to calculate Y^{-a}

$$Y^{p-1-a} = 1430^{605} \pmod{2357} = 872$$

$$m = Y^{-a} \cdot f \pmod{2357} = 872 \cdot 697 \pmod{2357} \\ = 2035$$

Rabin Public Key Enc. Scheme

⊕ Very fast enc.

⊖ Very slow dec.

Key generation Alg.

- ① Generate 2 large primes p, q nearly same size.
- ② Calc. $n = pq$
- ③ Alice's public key is n , private key is (p, q)

Enc. Alg. (by Bob)

- ① Obtain Alice's authenticated public key n .
- ② Represent the msg as an int ' m ' in the finite field / msg space: $\{0, 1, \dots, n-1\}$
- ③ Cipher Text $C = m^2 \pmod{n}$. m^2 is much less than RSA's m^e .
- ④ Send cipher text.

Dec. Alg (by Alice)

- ① Use an algorithm (3 provided later) to find 4 square roots m_1, m_2, m_3, m_4 of the cipher text mod n .
- ② The actual method m is one of m_1, m_2, m_3, m_4 .

e.g. $\overbrace{p=277 \quad q=331}^{\text{Private key}} \quad \overbrace{n=91687}^{\text{Public key}}$

$\longleftarrow 10 \longrightarrow$

10 bit msg: $m' = 101111001$

Add some redundant info to make it 16 bits and in order to help the receiver to recognize which is the actual msg. We repeat the last 6 bits. So we send:

$\longleftarrow 16 \longrightarrow$

$m = 1001111001111001$

Receiver will extract 4 candidates for the msg. and this repetitiveness will help him find the correct one.

- Convert to decimal $m = 40569$
 - Enc. $C = m^2 \pmod{n} = 40569^2 \pmod{91687} = 62111$
 - Dec. Find 4 square roots of $62111 \pmod{n}$
- $m_1 = 69654 \quad m_2 = 22033 \quad m_3 = 40569 \quad m_4 = 51118$

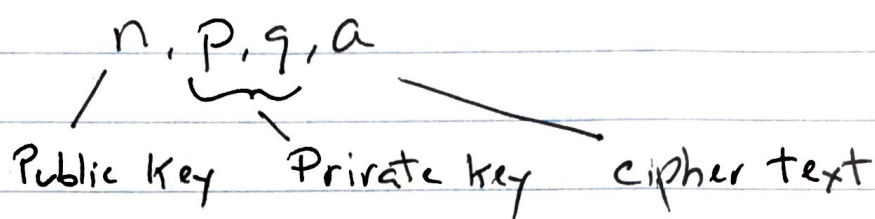
Can verify each: $m_x^2 \pmod{91687} \equiv 62111$ Ok ✓

Convert to bin: $m_1 = 10001000000010110$
 $m_2 = 101011000010001$
 $m_3 = 1001111001111001$
 $m_4 = 1100011110101110$

Receiver will recognize this as the real msg because of the repetition. $m_3 = 40569$

Calculate square roots mod n of its prime factors p, q

input:



Output: 4 square roots of 'a' mod n.

4 sq. roots instead of 2 bcuz the $\sqrt{}$ of a no. returns 2 values, one pos., one neg.

① Use another alg. X to find 2 sq. roots r and $-r$ of 'a' mod (p)

② Use X again to find 2 more sq. roots, call them s and $-s$ of 'a' mod (q)

0:24 ③ Use EE Alg to find $CP + dq = 1$ (find c and d)

④ calc:

$$x = rdq + scp \pmod{n}$$

$$y = rdq - scp \pmod{n}$$

⑤ Return $(\overset{\pm}{x}, \overset{\pm}{y}) \pmod{n}$

F is $\pm$

- purpose was to find sq. root mod n ,
not sq. rt. mod p .

e.g. cont.

Cipher text = 'a' = 62111

$$p = 277$$

$$q = 331$$

$$n = 91687$$

2 cont. assume we

already have:

$$r = 150 \quad -r = -150$$

$$s = 144 \quad -s = -144$$

Find C and d

3 cont. Use EE. Alg. to find: $qd + pc$

$$(331)(118) + (277)(-141) = 1$$
$$q \cdot d + p \cdot c = 1$$

4 cont.

$$x = rdq + scp \pmod{n}$$

$$y = rdq - scp \pmod{n}$$

find $\pm x, \pm y$

$$x = rdq + -scp \pmod{n}$$

$$= (150)(118)(331) + (-144)(-141)(277) \pmod{91687}$$

$$= 11,482,908 \pmod{91687} = \boxed{22033 = m_2}$$

$$-x = -22033. \text{ Add } n \text{ until pos.} \rightarrow \boxed{69654 = m_1}$$

$$y = rdq - scp \pmod{n}$$

$$= (150)(118)(331) - (-144)(-141)(277) \pmod{91687}$$

$$\boxed{51118 = m_4}$$

$$-y = -51118 + n = \boxed{40569 = m_3}$$

-Convert all 4 to bin. Look for repetition

There are 3 variations on the X alg.

- 2 of them will work on only certain primes or primes w. specific properties.
- 1 is very general
- 3rd one is out of scope.

X₁ Finds sq. rts. of $a \pmod p$
works if: $p \equiv 3 \pmod 4$

Input: an odd prime p where $p \equiv 3 \pmod 4$ and a .
So we need to be aware of this at the key gen. phase.

Output: 2 sq. rts of $a \pmod p$

1. Compute r . $r = a^{\frac{(p+1)}{4}} \pmod p$ // use s+m

2. return $(r, -r) \rightarrow (\pm r)^2 \pmod p \equiv a$

e.g. $q=331$ check $331 \equiv 3 \pmod 4$ OK ✓

Ciphertext $a^{\frac{(p+1)}{4}} \pmod q = 62111^{\frac{332}{4}} \pmod{331} = 144$

return ± 144

X₂ Find sq. rts mod prime p .
if $p \equiv 5 \pmod{8}$

Input: an odd prime p where $p \equiv 5 \pmod{8}$ and 'a'

Output: 2 sq. rts of 'a' $\pmod{p}$

① Compute d . $d = a^{\frac{p-1}{4}} \pmod{p}$

② if $d = 1 \xrightarrow{\text{then}} r = a^{\frac{p+3}{8}} \pmod{p}$

③ if $d = p-1 \longrightarrow r = 2a(4a)^{\frac{p-5}{8}} \pmod{p}$

④ return $(r, -r)$

e.g. $p = 277 \xrightarrow{\text{verify}} 277 \equiv 5 \pmod{8}$

calc d . $d = \text{cipherText 'a'} = 62111^{\frac{276}{4}} \pmod{277} = 276$

276 is $= p-1$, so execute line 3, not 2.

$r = (2 \cdot 62111)^{\frac{272}{8}} (4 \cdot 62111) \pmod{277} = 150$

$r = \pm 150$

Not on exam

General alg. - find sq. rts. 'a' prime p
w/o. any condition on p.

Input: odd prime p and 'a'

Output: 2 sq. rts. of 'a' (mod p)

① chose random $b \in \mathbb{Z}_p$ until $b^2 - 4a$ is
a quadratic non-residual (mod p)

$$\left(\frac{b^2 - 4a}{p} \right) = -1$$

② let f be a polynomial $(x^2 - bx + a) \in \mathbb{Z}_p$

③ compute r. $r = x^{\frac{p+1}{2}} \pmod{f}$

④ return (r, -r)

all coefficients are
in finite field $\mathbb{Z}_p$

10/11

Blum - Goldwasser - a probabilistic public key enc. scheme. Probabilistic coz at some point, they make a random decision.

Basic security definition: a minimal security requirement of an enc. scheme is that it must be difficult in essentially all cases for a passive adversary (plays follow protocols, but are curious to learn the secret) to recover the plaintext from the ciphertext.

Deterministic enc. schemes - RSA, Rabin

- The random primes in RSA, etc. don't count as a randomized alg.
- Under a fixed public key, a particular plaintext 'm' is always enc. to the same ciphertext.
 - easy to detect if a msg is sent twice
 - this problem can be resolved in public key schemes by using probabilistic approaches.

Quadratic Residue mod n

$$n = 10 \longrightarrow \mathbb{Z}_{10} \longrightarrow \overset{\text{non-0 el. of } \mathbb{Z}_{10}}{\{1, \sim 9\}}$$

0:17:00

$$1^2 \equiv_{10} \textcircled{1} \quad 2^2 \equiv_{10} \textcircled{4} \quad 3^2 \equiv_{10} \textcircled{9} \quad 4^2 \equiv_{10} \textcircled{6}$$

$$5^2 \equiv_{10} \textcircled{5} \quad 6^2 \equiv_{10} 6 \quad 7^2 \equiv_{10} 9 \quad 8^2 \equiv_{10} 4 \quad 9^2 \equiv_{10} 1$$

1, 4, 5, 6, 9 are quadratic residue mod 10

2, 3, 7, 8 are missing, these are non-quadratic residue mod 10

- a qqr is a perfect square of a random no. mod n .

10/11 0:17

Blum-Goldwasser - Key Gen

- ① Select 2 large random distinct primes p and q s.t. they are congruent to 3 mod 4.
 $p \equiv 3 \pmod{4}$ and $q \equiv 3 \pmod{4}$
- ② Compute n . $n = pq$ // security relies on // integer factors.
- ③ Use EE Alg. to compute a and b .
 $ap + bq = 1$. For sure we can do EE Als because p and q are prime no. They are coprimes, so $\gcd(p, q) = 1$
- ④ Public key is n .
private key is (p, q, a, b)

Blum - Goldwasser - Enc.

- a) obtain Alice's authenticated public key n .
b) calc. k . $k = \lfloor \log_2 n \rfloor$ // k denotes the no. bits in n .

calc. h . $h = \log_2 k$ // no. bits in h

- Represent msg. as a string $m_1, m_2, \dots, m_t$ where each m_i is a bin string of size h .

- c) Randomize behavior occurs in this step:

Select a seed X_0 (random quadratic residue mod n).

How to generate random quadratic residue?

Select a random no. r from $\mathbb{Z}_n^*$ $r \in \mathbb{Z}_n^*$

then calc. $r^2 \pmod n = X_0 = \text{random qr.}$

• a qr is a perfect square of a random no. mod n . X_0 varies the cipher text.

- d) Repeat 1..t // $t = \text{total no. substrings.}$

d₁) calc X_i . $X_i = X_{i-1}^2 \pmod n$ // $X_i = \text{seed}$ 10/11
27/17

d₂) p_i will be the " h " least significant bits of X_i

d₃) Calc: C_i // part of ciphertext.

from random process

$$C_i = p_i \oplus m_i$$

/ \
part of ciphertext part of msg.

- e) $X_{t+1} = X_t^2 \pmod n$ // calculate one more x after the loop in step d₃

f) ciphertext = $(C_1, C_2, \dots, C_t + X_{t+1})$

Blum - Goldwasser - Dec.

a) calc. d_1 . $d_1 = \left(\frac{p+1}{4}\right)^{t+1} \pmod{p-1}$

b) calc. d_2 . $d_2 = \left(\frac{q+1}{4}\right)^{t+1} \pmod{q-1}$

c) calc. u . $u = x_{t+1}^{d_1} \pmod{p}$ // x_{t+1} from sender

d) calc. v . $v = x_{t+1}^{d_2} \pmod{q}$

e) $x_0 = vap + ubq \pmod{n}$

f) $1..t$

comp. $x_i = x_{i-1}^2 \pmod{n}$ // if you have x_0

// you can get x_i

// same x_i sender gen.

calc. p_i = "h" least significant bits of x_i

compute $m_i = p_i \oplus c_i$

e.g. $p = 499$ $q = 547$ $11 \equiv 3 \pmod{4}$

$n = pq = 272953$ // reveal

Calc a, b using EEA. $(-57)(499) + (52)(547) = 1$

$$\underbrace{\begin{matrix} a & p & b & q \end{matrix}}_{\text{private key}}$$

Enc:

$k = \lfloor \log_2 n \rfloor = 18$

$h = \lfloor \log_2 k \rfloor = 4$ // if we assume the msg has 20 bits, every chunk should be 4 bits.

So 20 bit msg is m_1, m_2, m_3, m_4, m_5

assume:

$m_1 = 1001$ $m_2 = 1100$ $m_3 = 0001$ $m_4 = 0000$ $m_5 = 1100$

Sender gen. random no. X_0 . $X_0 = 359^2 \pmod{n} = 159201$ grg

loop: i	$X_i = X_{i-1}^2 \pmod{n}$	P_i	$C_i = P_i \oplus m_i$
1	* 180539 $\rightarrow$ bin	1011	0010 = 1011 $\oplus$ 1001
2	193932	1100	0000
3	245613	1101	1100
4	130286	1110	1110
t= 5	40632	1000	0100

- exit from loop and calc $X_6 = X_5^2 \pmod{n} = 139680$
- * get 4 least significant bits

$\rightarrow 0010, 0000, 1100, 1110, 0100, X_6$

Dec:

$$d_1 = \left(\frac{P+1}{4} \right)^6 \pmod{p-1} = 463$$

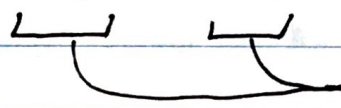
$$d_2 = \left(\frac{q+1}{4} \right)^6 \pmod{q-1} = 337$$

$$u = x_b^{463} \pmod{p} = 20$$

$$v = x_b^{337} \pmod{q} = 24$$

// x_b is public, sent w. cipher text.

$$x_0 = v a p + u b q \pmod{n} = 159201$$

 receiver already had, its part of private key

- calc. x_i 's and p_i 's

- From x_0 gen. x_i . check least sig. bits of x_i to get $p_i, \dots$

- When you have all the p_i 's, calc $m_i = P_i \oplus C_i$
↑
chunks