RSA Encryption

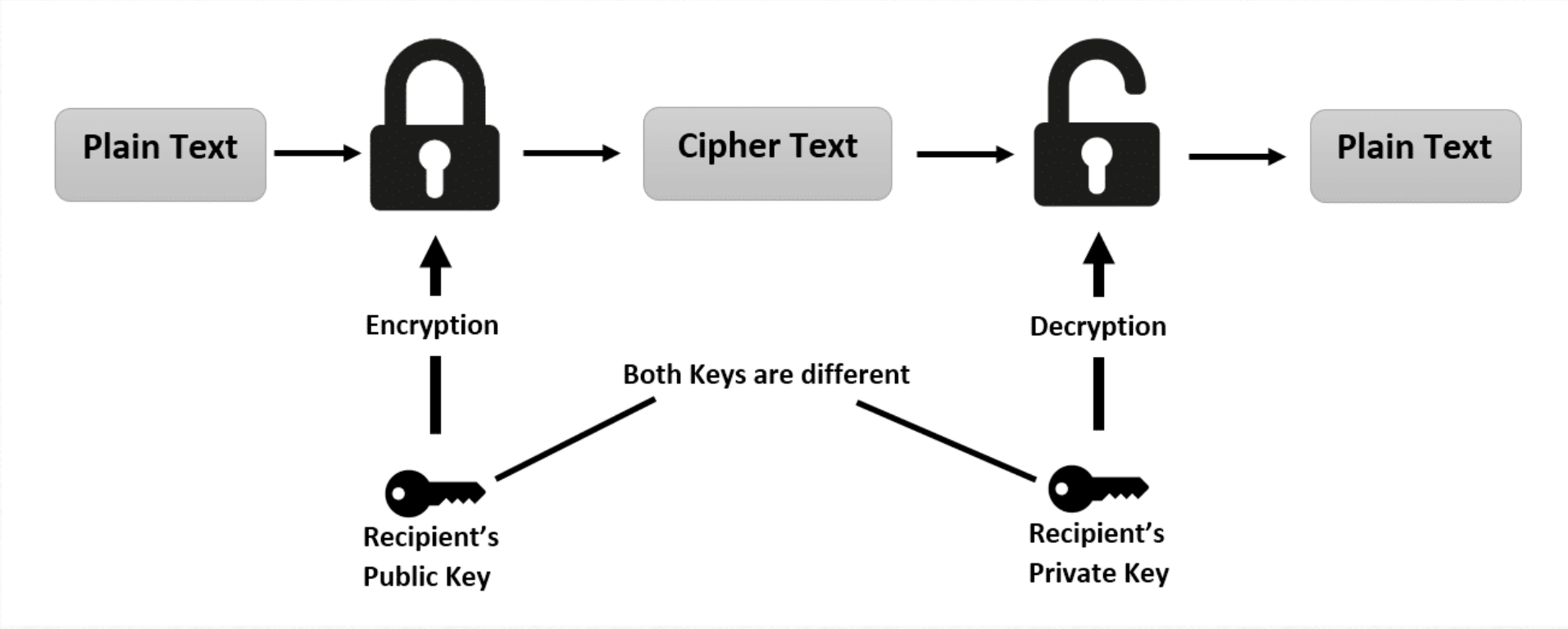
420-N320LA Networking and Security

Presented by: Omid Panahi

All About RSA

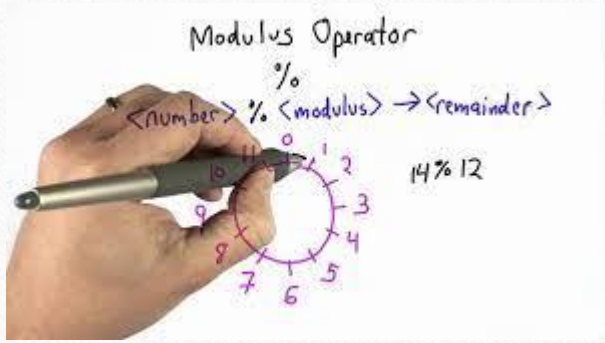
- The RSA algorithm is the most widely used Asymmetric Encryption algorithm deployed to date.
- The acronym is derived from the last names of the three mathematicians who created it in 1977: Ron Rivest, Adi Shamir, Leonard Adleman.

The Basic Idea (Asynchronous Encryption)



Modulus (%)

- This is a way of simply asking for a remainder.
- If presented with the problem 12 MOD 5, we simply are asking for the remainder when dividing 12 by 5, which results in 2.



Find Modulo with your Calculator

Method 1 (Simple Method)

1. 127 mod 24
2. 127 divided by 24 is 5.292, keep the whole number 5.
3. Take the number 24 and multiply by the number you found: 24 * 5 = 120.
4. Subtract what you found, 120, from the original number 127, this is your answer. : 127 – 120 = 7
5. Therefore 127 mod 24 = 7

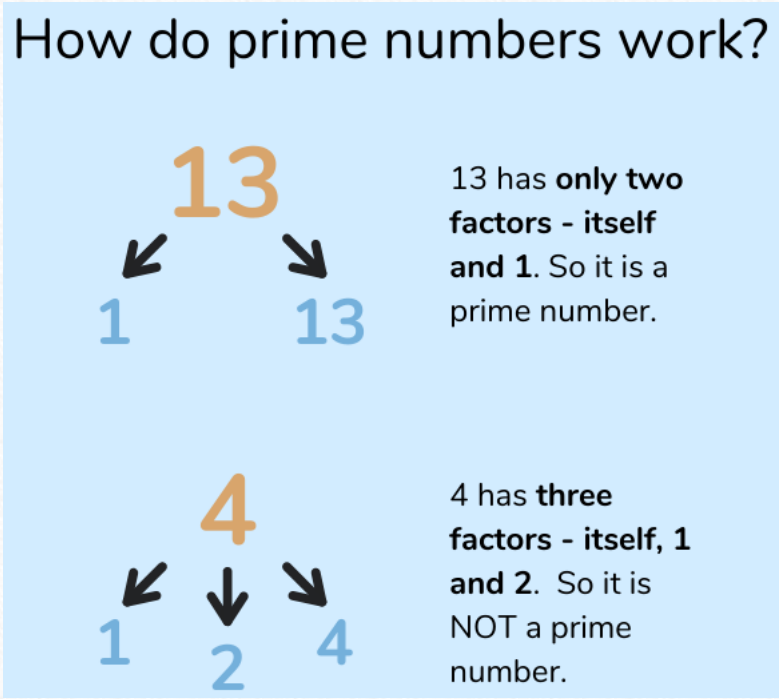
Method 2

1. 127 mod 24
2. 127 divided by 24 is 5.2916666666666666666666666666667, SUBTRACT the whole number 5.
3. Multiply the result (.2912) by the mod denominator 24.
4. .2917… * 24 = 7

There is a way to get modulus of REALLY big numbers, it's called "CRT" (Chinese Remainder Theory)

Prime Number

- A number is Prime if it is only divisible by 1 and itself. Such as: 2, 3, 5, 7, 11, 13, etc.



Code to check for a Prime Number

```
int n = 5, a = 0;

for (int i = 1; i <= n; i++) {
    if (n % i == 0) {
        a++;}
}

if (a == 2) {
    Console.WriteLine("{0} is a Prime Number", n);
} else {
        Console.WriteLine("Not a Prime Number");
}
```

Factor

- A factor is a number you can multiple to get another number. For example, the factors of 12 are 1, 2, 3, 4, 6, and 12.

Semi-Prime Number

- A number is Semi Prime if its only factors are prime (excluding 1 and itself). For example:
  - 12 is not semi-prime — one of its factors is 6, which is not prime.
  - 21 is semi-prime — the factors of 21 are 1, 3, 7, 21. If we exclude 1 and 21, we are left with 3 and 7, both of which are Prime.

As a hint: Anytime you multiply two Prime numbers, the result is always Semi Prime

Coprime

- Co-prime numbers are pairs of numbers that do not have any common factor other than 1.
- There should be a minimum of two numbers to form a set of co-prime numbers. Such numbers have only 1 as their highest common factor, for example, (4 and 7), (5, 7, 9) are co-prime numbers.
- It is to be noted that co-prime numbers need not be prime numbers always. Two composite numbers like 4 and 9 also form a pair of co-primes.

https://www.cuemath.com/numbers/coprime-numbers/

How to verify a co-prime

RSA Procedure

Select Prime Numbers

- Select two prime numbers, p and q.
- Larger prime numbers are easy to work with on the compute, select lower values when working on paper.
- Select two prime numbers to begin the key generation. For this example, we will use the numbers 7 and 19, (p and q respectively).

Calculate Product

- Multiply p and q, the product and assign this to "n".
- We will refer to this number as N
- given the terminology we reviewed above, what kind of number is N?

Calculate the Totient of N

- This is done by calculating: (P-1)*(Q-1)
- We will simply accept that the formula to attain the Totient on a Semi Prime number is to calculate the product of one subtracted from each of its two prime factors. Or more simply stated, to calculate the Totient of a Semi-Prime number, calculate P-1 times Q-1.
- Let's call this: φ(n) (phi n) or T

Select a public Key

- The Public Key is a value which must match three requirements:
  - It must be Prime
  - It must be less than the Totient
  - It must NOT be a factor of the Totient
- Let's try number 3: 3 is indeed Prime, 3 is indeed less than 108, but regrettably 3 is a factor of 108, so we can not use it. Can you find another number that would work? Here is a hint, there are multiple values that would satisfy all three requirements.
- We will select 29 as our Public Key known as E

Select a Private Key

- We can now select our Private Key (which we will call D).The Private Key only has to match one requirement:
  - (D*E) MOD T = 1
  - The Product of the Public Key and the Private Key when divided by the Totient, must result in a remainder of 1.
- There are a few values that would work for the Private Key as well. But again, for the sake of our example, we will select 41. To test it against our formula, we could calculate:
  - (41*29) MOD 108
  - = 1

How to Find D?

D=E^{-1} mod ϕ, also written d=(1/E) mod ϕ

D= E^{-1} mod $\varphi(n)$

D=29^{-1} mod 108

D = 41

Procedure to Find $\mathbf{d=e^{-1} \bmod \varphi(n)}$

- For smaller numbers, this is not hard to do:
- For example: $d=3^{-1} \bmod 7$, substitute A=3, C=7, find B using this pattern:

1. A*B mod C = 1
2. 3*B mod 7 = 1
3. Choose numbers for B from 0 to C-1 until one works (warning, only for small values).
4. Brute Force Method for Small Numbers

Let's combine all calculation together

$p = 5, q = 11$ (These are both prime numbers)

Compute $n = p \times q$

- $n = 5 \times 11 = 55$
- n is part of both the **public** and **private** keys.

Compute the totient $\varphi(n)$:

- For RSA with two primes:
- $\varphi(n) = (p-1)(q-1)$ → φ(n)=(5−1)(11−1)=4×10=40

- **Choose $e$such that**
- We need a number **e** that:
- $1 < e < \varphi(n)$
- Shares **no common factors** with 40 (other than 1)
- Let's try **e = 3**:
- divisors of 3 ⟶ 1, 3
- divisors of 40 ⟶ 1, 2, 4, 5, 8, 10, 20, 40
- common divisor ⟶ **only 1**
- So: Public key so far: **(e, n) = (3, 55)**
- Find $d = e^{-1} \bmod \varphi(n)$: We want **d** such that: $d \cdot e \equiv 1 \pmod{40}$

Check:(Brute Force)

$3 \times 10 = 30 \longrightarrow 30 \bmod 40 = 30$

$3 \times 20 = 60 \longrightarrow 60 \bmod 40 = 20$

$3 \times 27 = 81$

Now check $81 \bmod 40$:

$40 \times 2 = 80$

$81 - 80 = 1$

So:

$3 \times 27 \equiv 1 \pmod{40}$

$d = 27$

Private key: **(d, n) = (27, 55)**

The Different Values Being Used

Message Encryption

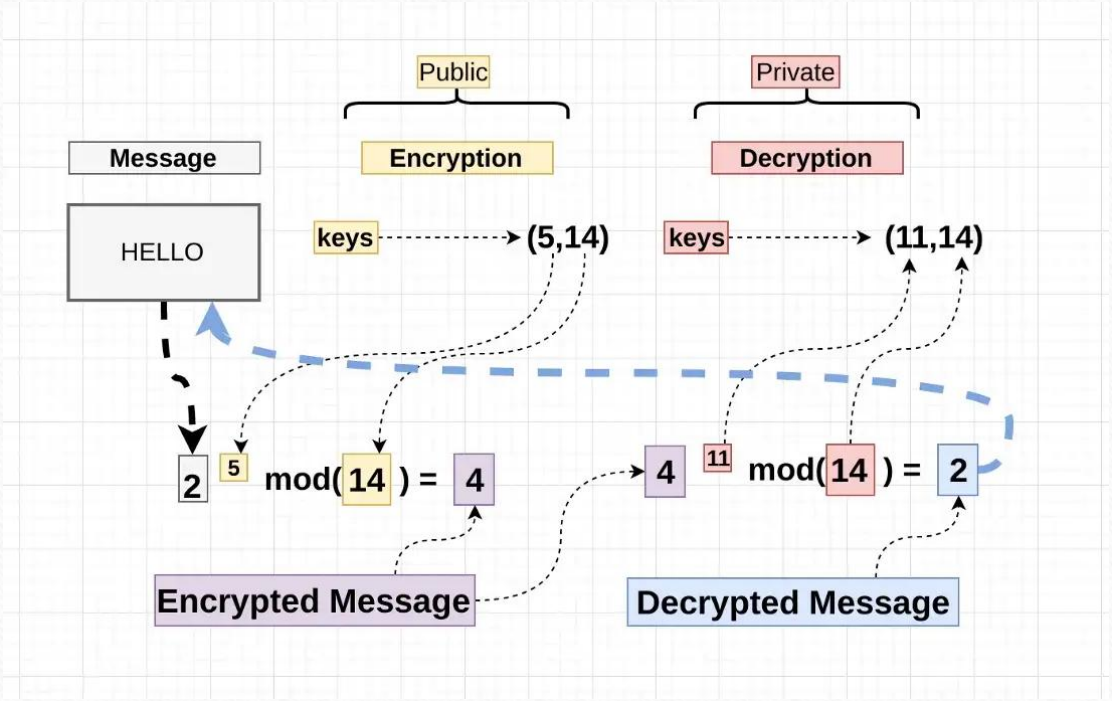
Setup of Encryption

- Recall, that with Asymmetric Encryption, we are encrypting with the Public Key, and decrypting with the Private Key.
- You'll notice that, as was stated before, it is impossible to use the same key to both encrypt and decrypt.
- Encryption Formula:
  - Cipher Text = M^E MOD N
  - 99^29 MOD 133 = 92

Decryption

- The formula to Decrypt with RSA keys is:
  - Original Message = M^D MOD N
  - 92^41 MOD 133 = 99

A Summary of RSA



A Mathematical Summary of RSA

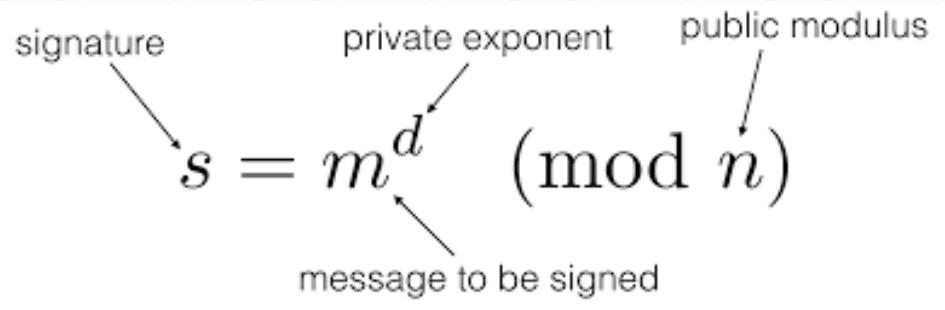
Key generation	$n = P * Q$ $d * e = 1 \bmod \Phi(n)$
Encryption	$c = m^e \bmod n$ Public Key(n,e)
Decryption	$m = c^d \bmod n$ private key (d)

Signing

RSA Digital Signing

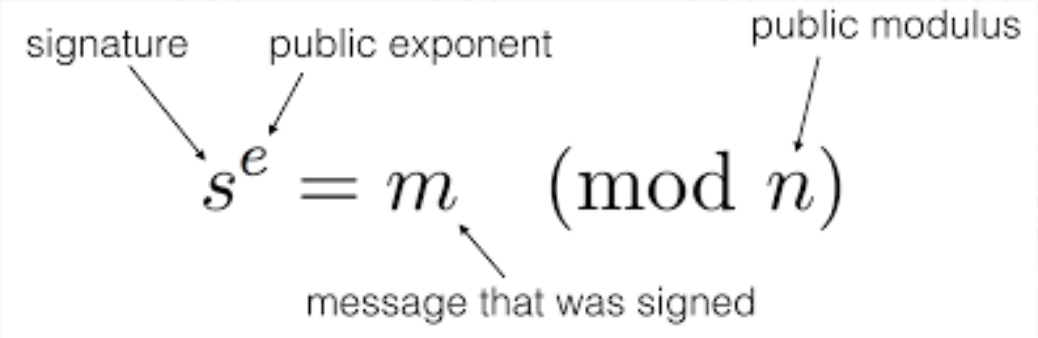
Signing

- Signing is the reverse as encryption.
- We can also use same key pair in the opposite order to verify a message's signature.
- We're going to encrypt with the Private Key and see if we can decrypt with the Public Key.
  - Signature = M^D MOD N
  - 99^41 MOD 133 = 36
- 36 is the Signature of the message.

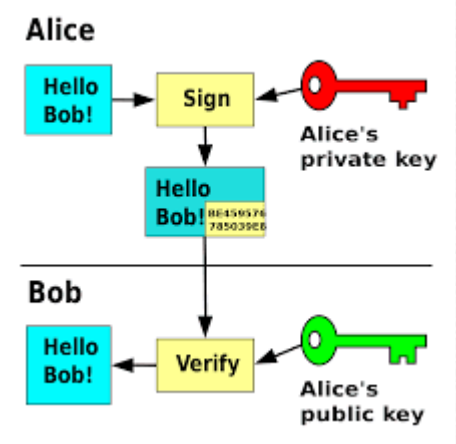


Signing Continued

- The result of 36 is the Signature of the message. If we can use the correlating public key to decrypt this and extract the original message, then we know that only whoever had the original Private Key could have generated a signature of 36.
- Verification of signature:
  - Original Message = M^E MOD N
  - 36^29 MOD 133 = 99



Signature


Alice
Hello Bob!
Sign
Alice's private key
Hello Bob!
Bob
Hello Bob!
Verify
Alice's public key