HFSJH PZKF BJGGNZFI WRVS PZKF HZZHEUFKGE. IZY'H

WSH JYPUZIP SWGS KGS RH, JYI ASH J YSN ZYS SMSFP

GRO XZYHEG. -LWRCCZFI GHZWW

TREAT YOUR PASSWORD LIKE YOUR TOOTHBRUSH. DON'T LET ANYONE ELSE USE IT, AND GET A NEW ONE EVERY SIX MONTHS. -CLIFFORD STOLL

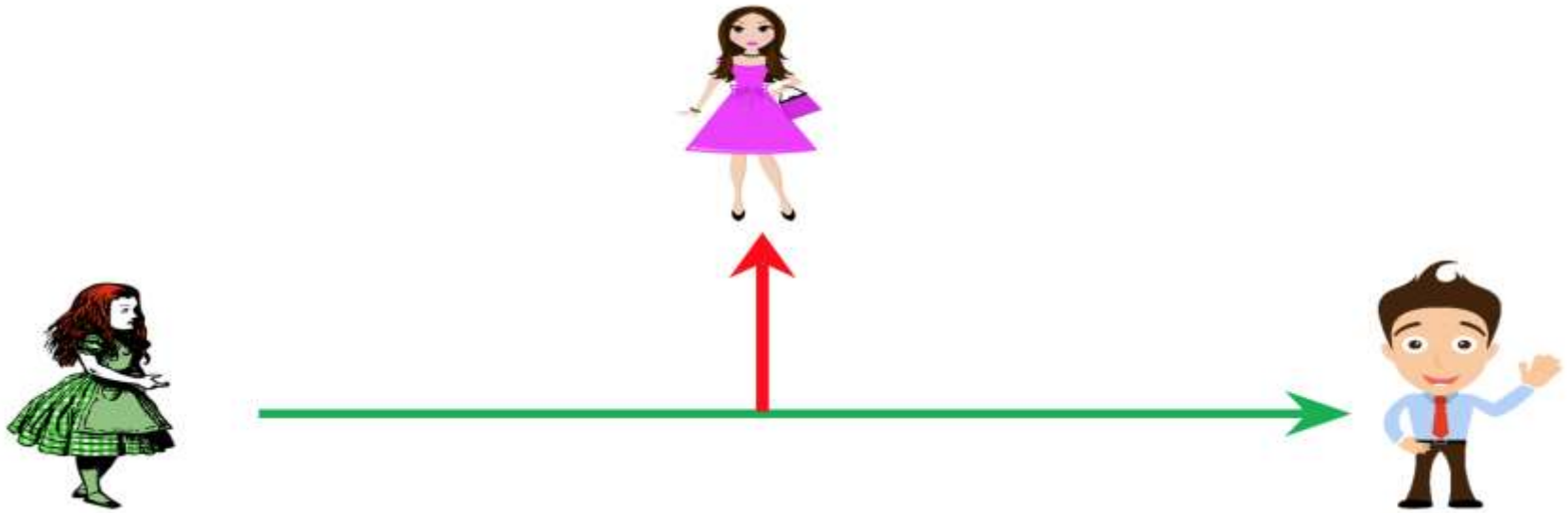
Computer Networks Lab(18CSL57)

5 Sem B and C section.
-Shanthala KV
-Shweta Kaddi

(18CSL57)

11. Write a program for simple RSA algorithm to encrypt and decrypt the data

- Consider two people in different locations, **Alice** and **Bob**, who want to communicate in secret
- A third person, **Eve**, wants to intercept that communication



Locked Box

- Alice and Bob share a combination to a locked box
- Alice and Bob can both open it, but Eve cannot.

Basic Terminology

- *plaintext* - original message
- *ciphertext* - coded message

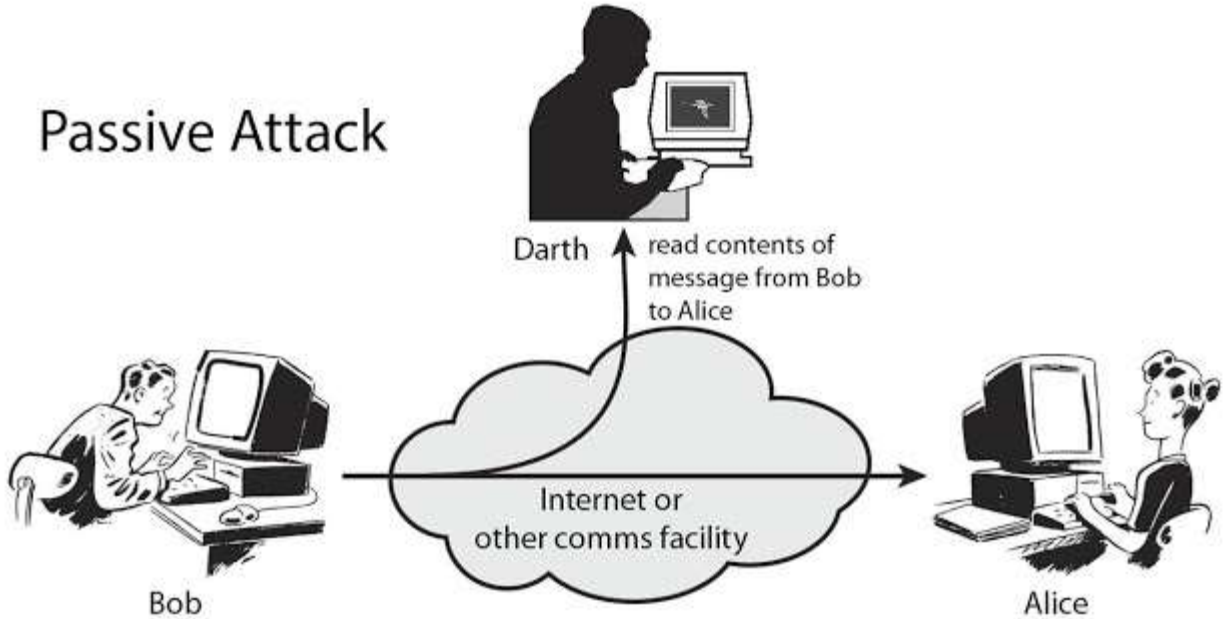
- *cipher* - algorithm for transforming plaintext to ciphertext
- *key* - info used in cipher known only to sender/receiver

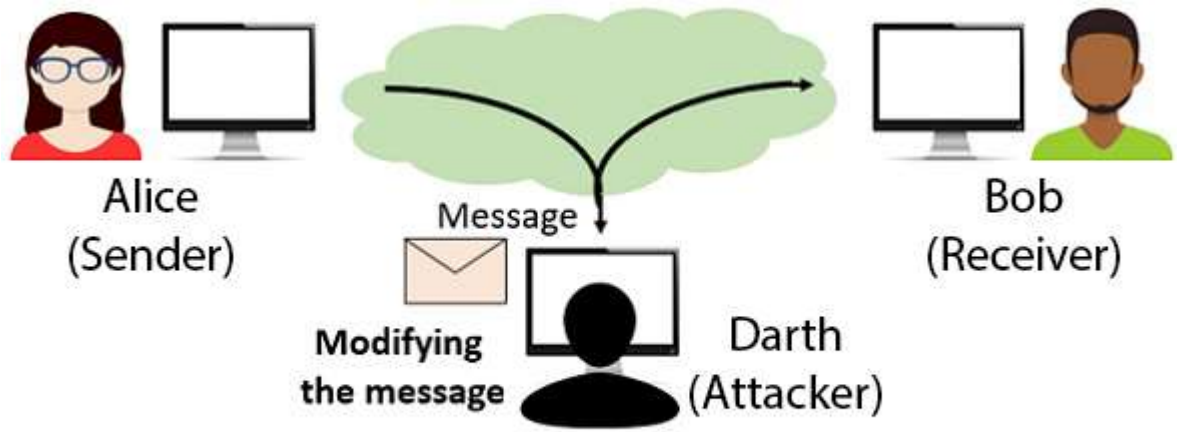
- *encipher (encrypt)* - converting plaintext to ciphertext
- *decipher (decrypt)* - recovering plaintext from ciphertext

- *cryptography* - study of encryption principles/methods
- *cryptanalysis (codebreaking)* - study of principles/methods of deciphering ciphertext without knowing the key

- *cryptology* - field of both cryptography and cryptanalysis

Passive Attack

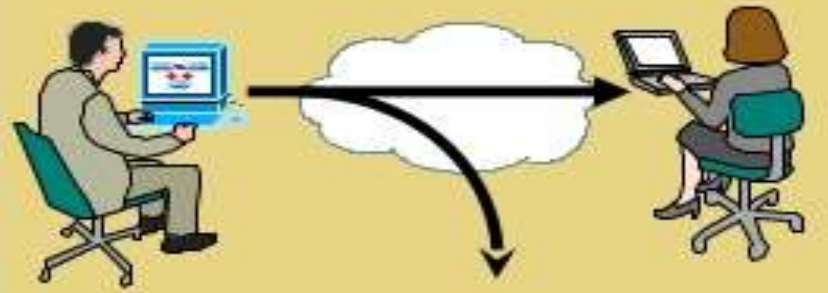




Active Attack

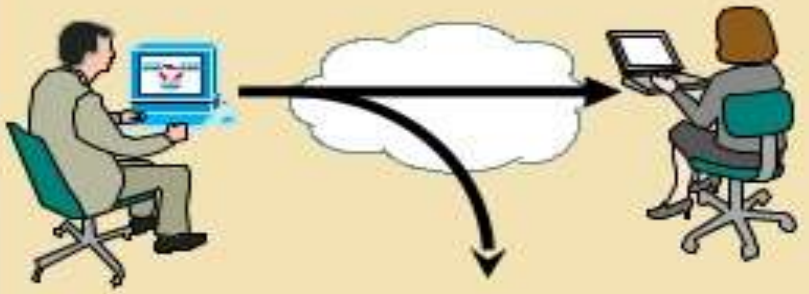
Passive Attack

Release of Message Contents



Read contents of message from Bob to Alice

Traffic Analysis

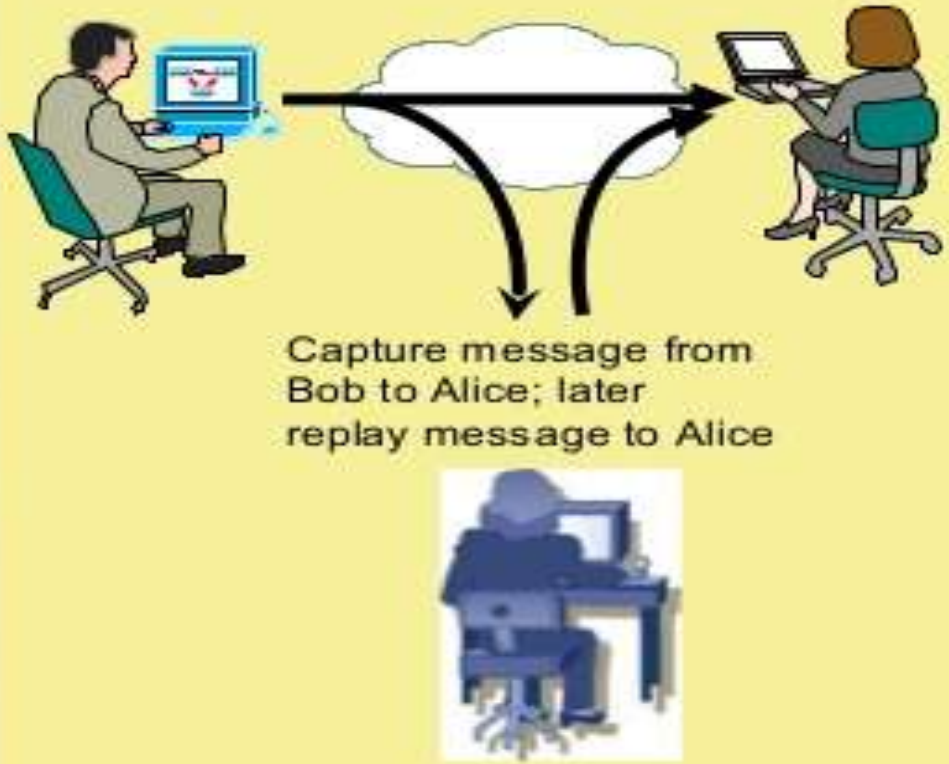


Observe pattern of messages from Bob to Alice

Active Attack - 1

Masquerade

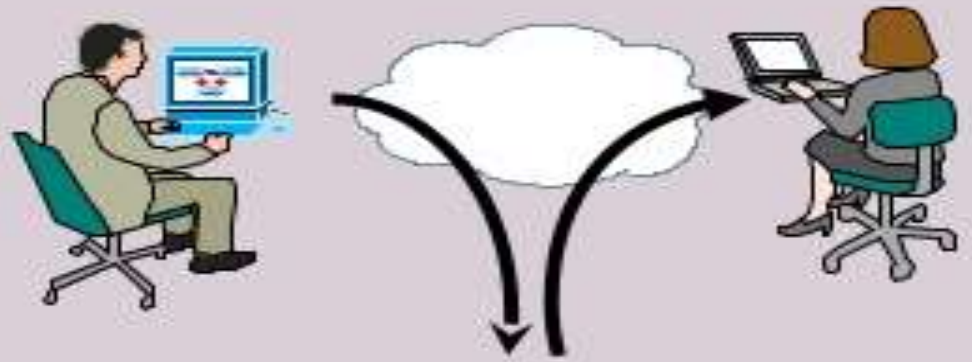
Replay*



* An attack in which a service already authorized and completed is forged by another "duplicate request" in an attempt to repeat authorized commands.

Active Attack - 2

Modification of messages



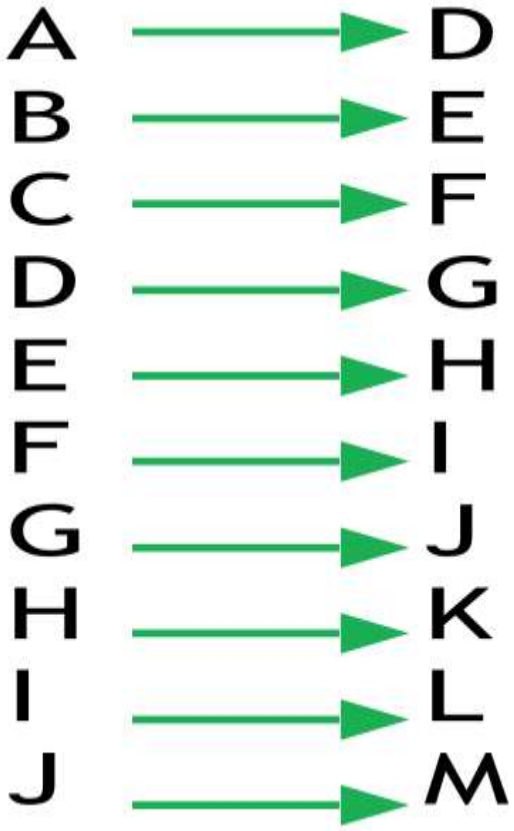
Modifies message from Bob to Alice

Denial of Service

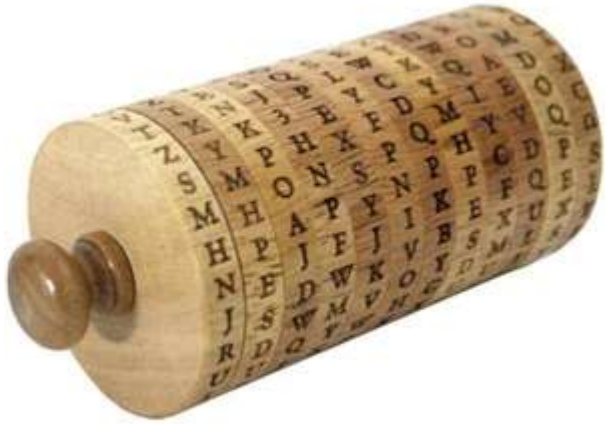
disrupts service provided by server

Caesar Cipher

- Shift (Rotate) letter by agreed-upon number
- Recipient shifts back by the same number of letters
- Used for hundreds of years after Caesar's time!



Simple Cipher Wheels



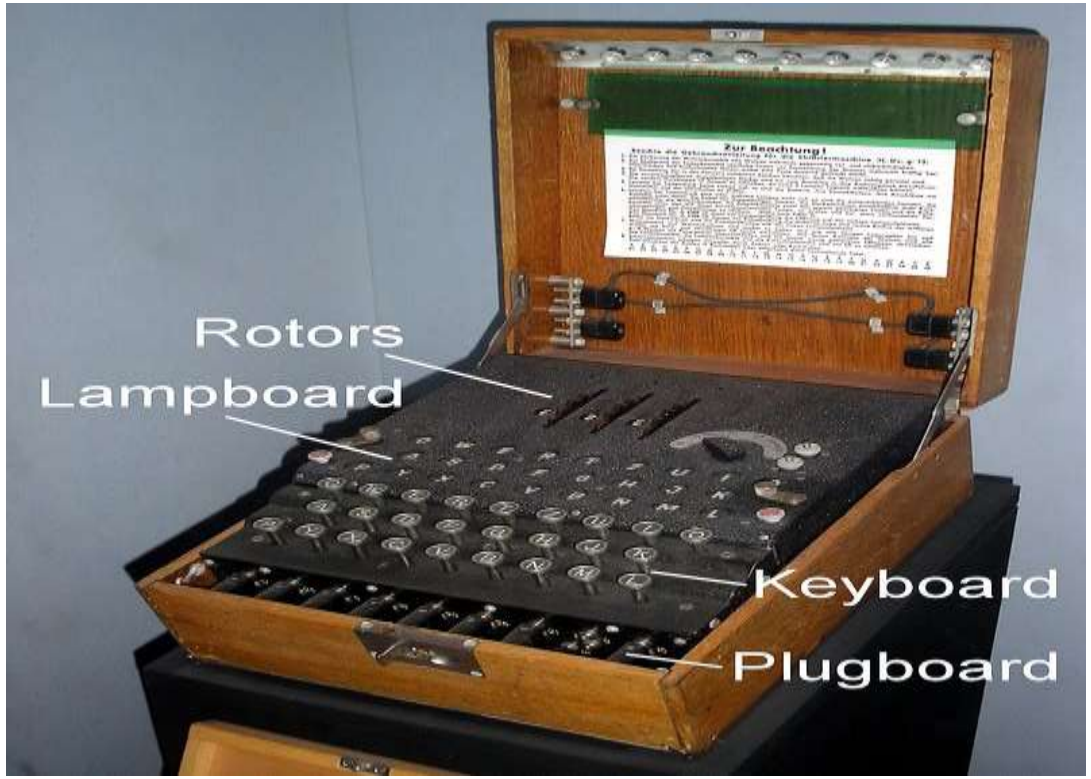
Row Transposition Cipher

- A more complex transposition
- Write letters of message in a rectangle, row by row
- Permute order of columns
- Read message off column by column

```
Key:         4 3 1 2 5 6 7
Plaintext:   a t t a c k p
             o s t p o n e
             d u n t i l t
             w o a m x y z
Ciphertext:  TTNAAPTMTSUOAODWCOIXKNLYPETZ
```

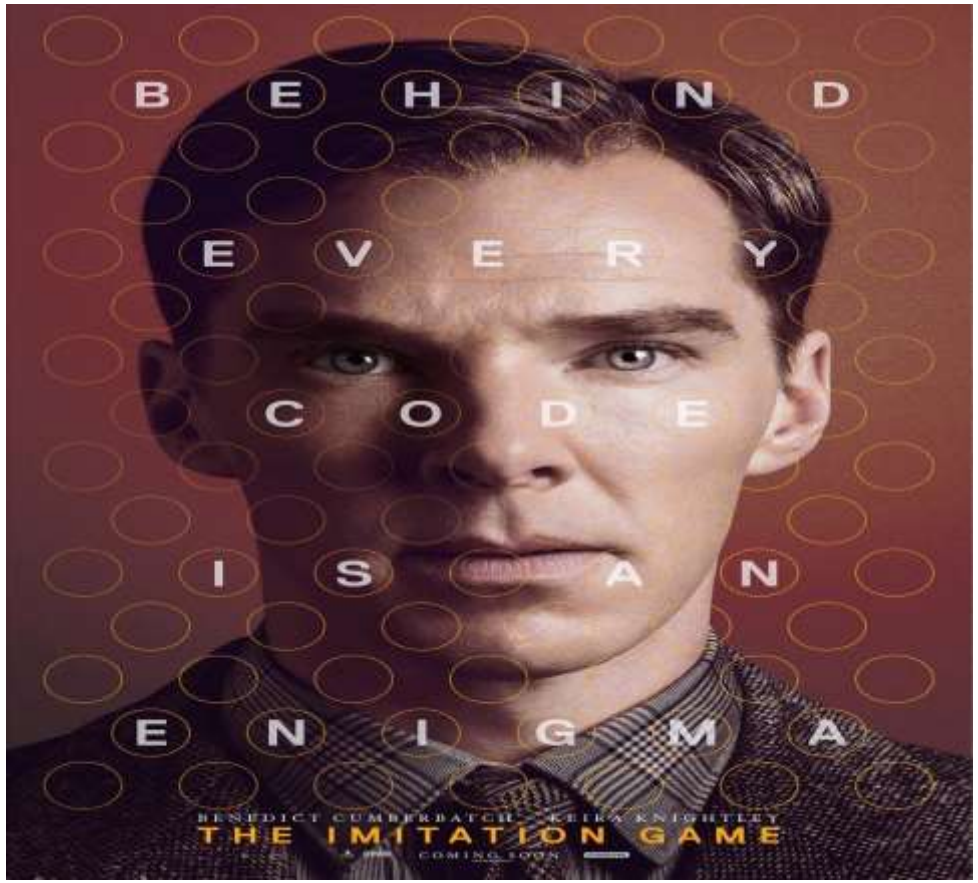
Rotor Machines

- Before modern ciphers, rotor machines were most common complex ciphers in use
- Widely used in WW2
  - German Enigma, Allied Hagelin, Japanese Purple
- Implemented a very complex, varying substitution cipher
- Used a series of cylinders, each giving one substitution, which rotated and changed after each letter was encrypted
- With 3 cylinders have 263=17576263=17576 alphabets



Alan Turing: British mathematician, codebreaker, and early computer scientist, Alan Turing (1912–1954)
Alan Turing: The Enigma The book covers Alan Turing's life and work.

Movie: ***The Imitation Game*** is loosely based on the book, with dramatization.



Untrusted Communication Channels

- This is a story about Alice and Bob.
- Alice wants to send a private message to Bob, and the only easy way they have to communicate is via postal mail.

- Unfortunately, Alice is pretty sure that the postman is reading the mail she sends.

- That makes Alice sad, so she decides to find a way to send messages to Bob without anyone else being able to read them.

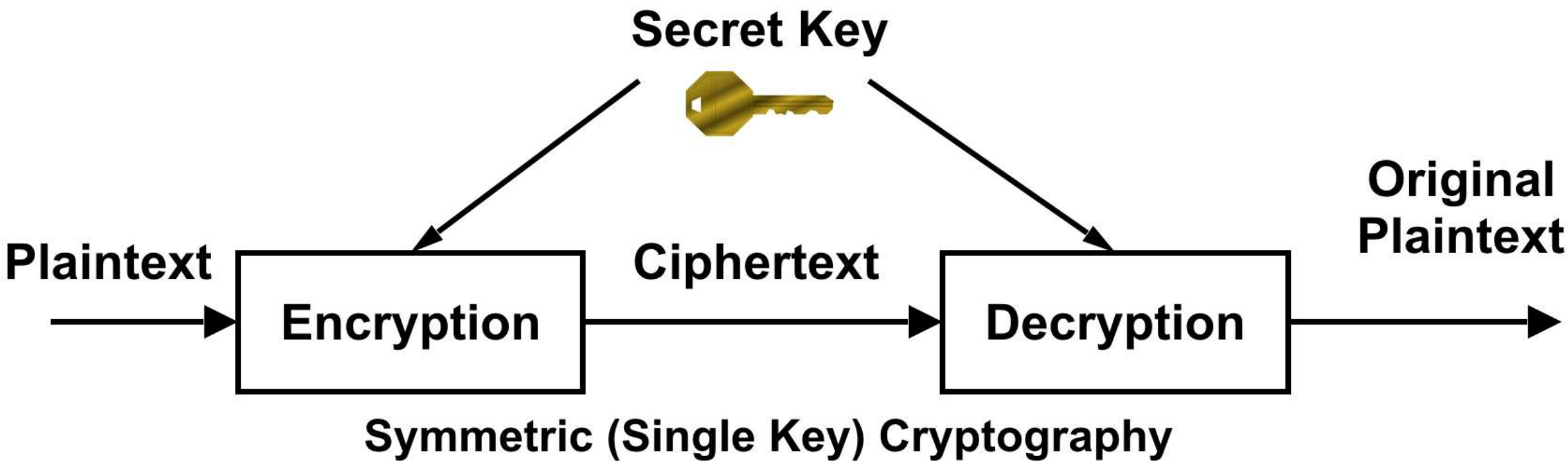
Symmetric-Key Encryption

- Alice decides to put the message inside a lockbox, then mail the box to Bob. She buys a lockbox and two identical keys to open it. But then she realizes she can’t send the key to open the box to Bob via mail, as the mailman might open that package and take a copy of the key.
- Instead, Alice arranges to meet Bob at a nearby restaurant to give him one of the keys. It’s inconvenient, but she only has to do it once.

- After Alice gets home she uses her key to lock her message into the lockbox.

Symmetric Cryptography

Eg: DES , Triple DES , AES, Blow Fish



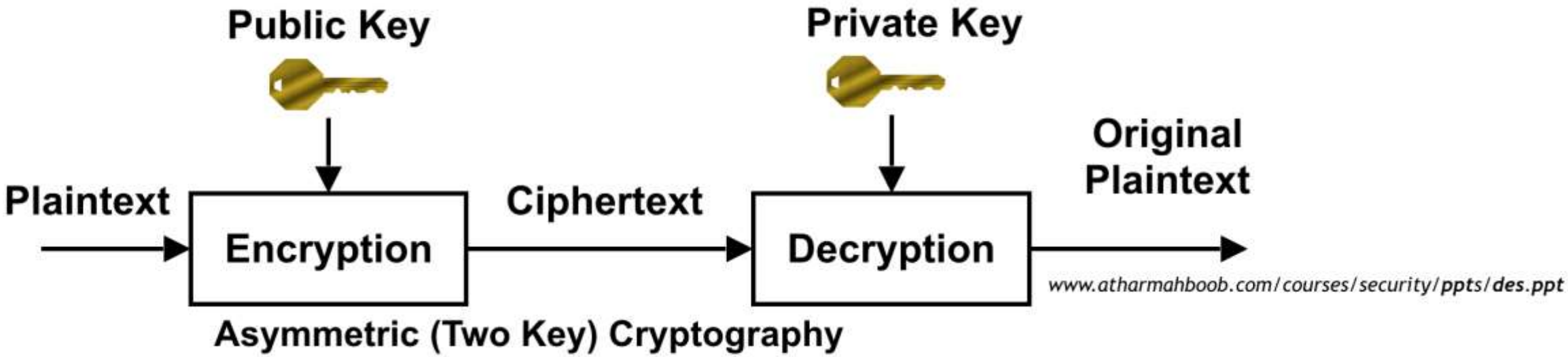
How large is a good key size?

- These days a good key size is: 256 binary bits (i.e., in the form of 1s and 0s)
- Why 256 bits?
  - In binary arithmetic, with 256 bits we can have a total of 22562256 possible choices for a key.
  - Assume an attacker can test a billion keys per second (= 109109 choices/second)
  - 22562256 keys will then take: 2256/1092256/109 ~ 1077/109=10681077/109=1068 seconds ~ 10161016 years!
  - 10161016 years is a lot of years!!
- So with a 256 bit key brute forcing through all the keys clearly takes a lot of time--for now!

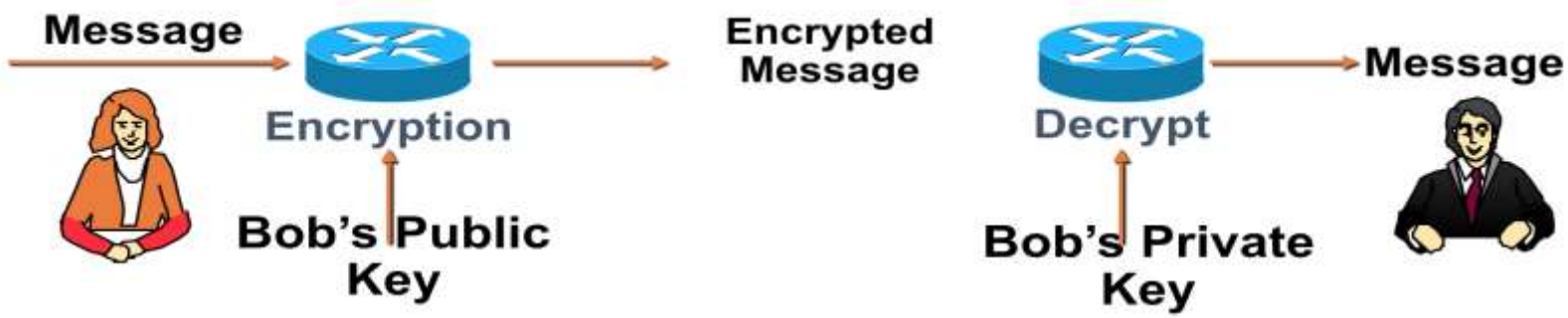
Disadvantages:

- Key Distribution
- Key Management

Asymmetric Cryptography



- We use different (but related) keys for encryption and decryption
  - 'Public key cryptography'
  - Sender encrypts with receiver's public key
  - Receiver decrypts with receiver's private key
- EXAMPLES: RSA Algorithm, Elliptic Curve Cryptography



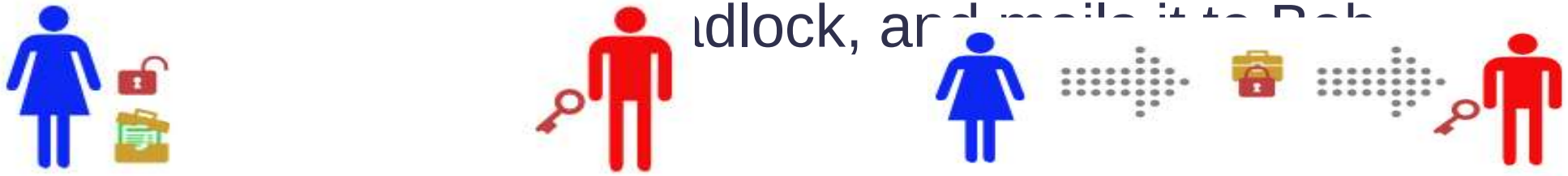
Public-Key Encryption /Asymmetric

- This time, Alice and Bob don't ever need to meet. First Bob buys a padlock and matching key

- Then Bob mails the (unlocked) padlock to Alice, keeping the key safe.

- Alice buys a simple box that closes with a padlock, and puts her message in it.

Then [illegible] padlock, and mails it to Bob



This only works to send messages in one direction

Solution

- Alice could buy a blue padlock and key and mail the padlock to Bob so that he can reply.
- Or, instead of sending a message in the padlock-secured lockbox, Alice could send Bob one of a pair of identica

- Then Alice and Bob can send messages back and forth in their symmetric-key lockbox, as they did in the first example.

RSA

- Proposed by Rivest, Shamir, and Adleman in 1977 and a paper was published in The Communications of ACM in 1978.
- Based on the idea that factorization of integers into their prime factors is hard.
- A public-key cryptosystem

RSA Steps:

1.Choose two primes p, q.

2.Compute n=p x q.

3. Z= (p-1) (q-1) (called Totient Function - ϕ)

4.Choose a random integer e, such that gcd (e,z) =1 [e and z are relatively prime].

5. {e,n} → Public Key

6. Calculate d such that de =1 mod z

7.{d,n} → Private key

8.Encrypt: $C=P^e \bmod n$

9.Decrypt : $P = C^d \bmod n$

Example :

1.p=7 q=11

2. n=7 x 11 =77

3. z=(7-1) x (11-1)= 60

4. Find e:

- such that e and z are relatively prime.
- Let e =13

5. Find d:

- Such that Z divides ed-1 (d= e^{-1} mod z)
- Here Z divides 13d-1
- d=37

Now Say I want to send 5 as the message.

6. C= 5^{13} mod 77 = 26 (C- Encrypted msg –Cipher Text)

7. P= 26^{37} mod 77 =5 (P- Plain Text)

Example 2:

Let Plaint text=18,19,1

1. p= 5 q= 11 n=55 z=40
2. Gcd (e,z) =1 → let e=7
3. Z divides ed-1 i.e, 40 divides 7d-1
4. d=?

5. $C1 = 18^{7} \bmod 55 = 17$
   $C2 = 19^{7} \bmod 55 = 24$
   $C3 = 1^{7} \bmod 55 = 1$

6. $P1 = 17^{23} \bmod 55 = 18$
   $P2 = 17^{23} \bmod 55 = 19$
   $P3 = 17^{23} \bmod 55 = 1$

- Bob chooses two primes p,q and compute n=pq
- Bob chooses e with gcd(e,(p-1)(q-1))= gcd(e, ψ(n))=1
- Bob solves $de \equiv 1 \pmod{\psi(n)}$
- Bob makes (e,n) public and (p,q,d) secret
- Alice encrypts M as $C \equiv M^e \pmod{n}$
- Bob decrypts by computing $M \equiv C^d \pmod{n}$

Some Maths Facts

FACT 1. Prime generation is easy: It's easy to find a random prime number of a given size

FACT 2. Multiplication is easy: Given p and q, it's easy to find their product, n = pq.

CONJECTURE 3. Factoring is hard: Given such an n, it appears to be quite hard and expensive to recover the prime factors p and q.
For instance, it has been estimated recently that recovering the prime factors of a 1024-bit number would take a year on a machine costing US $10 million.

FACT 4. Modular exponentiation is easy: Given n, m, and e, it's easy to compute $c = m^{\mathbf{e}} \bmod n$.

FACT 5. Modular root extraction – the reverse of modular exponentiation – is easy given the prime factors: Given n, e, c, and the prime factors p and q, it's easy to recover the value m such that $c = m^{e} \bmod n$.

CONJECTURE 6. Modular root extraction is otherwise hard: Given only n, e, and c, but not the prime factors, it appears to be quite hard to recover the value m.

Some more maths..

$de \equiv 1 \bmod Z$.

- The value d is the inverse of e modulo Z. The requirement that e be relatively prime to p1 and q-1 ensures that an inverse exists.

- Modular inverses are easy to find with the Extended Euclidean Algorithm or similar methods.

- The RSA cryptosystem works because exponentiation to the d^{th} power modulo n is the inverse of exponentiation to the e^{th} power when the exponents d and e are inverses modulo L. That is, for all m between 0 and n-1, $m \equiv (m^e)^d \bmod n$
.

```
import java.util.Scanner;
public class Main
{
public static void main(String[] args)
{
Main rsa=new Main();
int p = 0,q = 0;
Scanner in=new Scanner(System.in); System.out.println("Enter the message to be encrypted: "); String
message=in.nextLine();
char[] msgA=message.toCharArray(); boolean check=false;
while(!check)
{
System.out.println("Enter two prime numbers: "); p=in.nextInt();
q=in.nextInt();
if(checkPrime(p) && checkPrime(q)) check=true;
}
int n=p*q;
int z=(p-1)*(q-1); int e = 0;
for(int i=2;i<z;i++)
{
if(rsa.gcd(i,z)==1)
{
e=i; break;
}
}
System.out.println("The value of e: "+e); System.out.println("Public key: ("+e+", "+n+")"); int d=0;
for(d=2;d<z;d++)
if((e*d)%z==1)
break; System.out.println("d: "+d); int ch;
```

```
int[] a=new int[message.length()]; int[] c=new int[message.length()]; int t=0;
for(int i=0;i<message.length();i++)

{
ch=(int)msgA[i]; a[i]=ch;
t=1;
for(int j=0;j<e;j++)
t=(t*ch)%n;
c[i]=t;
}
System.out.println("Message ASCII Cipher"); for(int i=0;i<message.length();i++)
System.out.println(" "+msgA[i]+"\t "+a[i]+"\t "+c[i]); char cha[]=new char[message.length()];
System.out.println("Decrypting...");
for(int i=0;i<message.length();i++)
{
ch=c[i]; t=1;
for(int j=0;j<d;j++)
t=(t*ch)%n; cha[i]=(char)t;
}
System.out.println(new String(cha));
}
public int gcd(int a,int b)
{
if(b==0)
return a; return gcd(b,a%b);
}
public static boolean checkPrime(int num1)
{
if(num1==0 || num1==1)
return false; else if(num1==2)
return true; for(int i=2;i<num1;i++)
if(num1%i==0)
return false;
return true;
}
}
```

$$C=P^e \bmod n$$

$$P = C^d \bmod n$$

An Important Note

Encrypt: $C=P^e \bmod n$

Decrypt : $P = C^d \bmod n$

$$P = (P^e \bmod n)^d \bmod n$$

$$P = P^{ed} \bmod n$$

$$P = P \bmod n$$

For this to happen $P < n$ (P is the plain text)

i.e., $P < p \times q$ (product of prime numbers)

GCD:

idea of the algorithm is based on the observation that, if *r* is the remainder when *a* is divided by *b*, then the common divisors of *a* and *b* are precisely the same as the common divisors of *b* and *r*. Thus, we can use the equation.

GCD(a, b)= GCD(b, r)

$$\begin{aligned} \mathrm{GCD}(206, 40) &= \mathrm{GCD}(40, 6) \\ &= \mathrm{GCD}(6, 4) \\ &= \mathrm{GCD}(4, 2) \\ &= \mathrm{GCD}(2, 0) \\ &= 2 \end{aligned}$$

```
public int gcd(int a, int b)
{
if(b==0)
return a; return gcd(b, a%b);
}
```