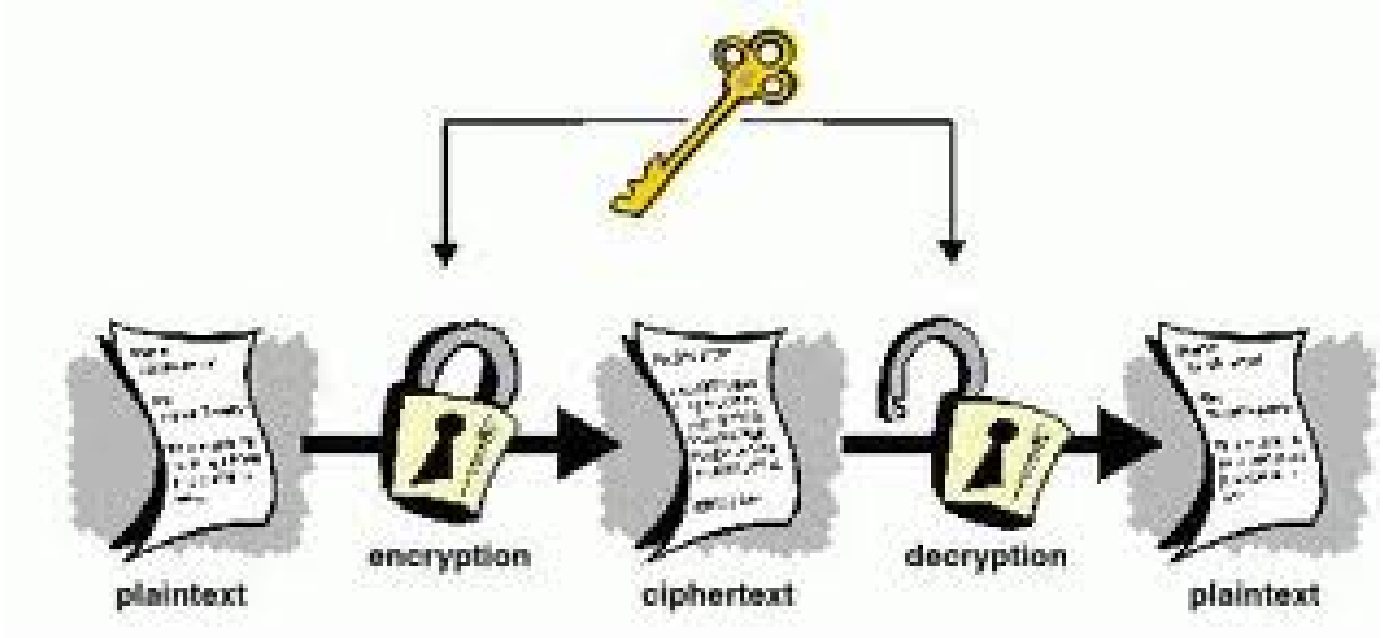
EECS 3482
Introduction to Computer Security



Instructor: N. Vlajic, Fall 2021

Learning Objectives

Upon completion of this material, you should be able to:

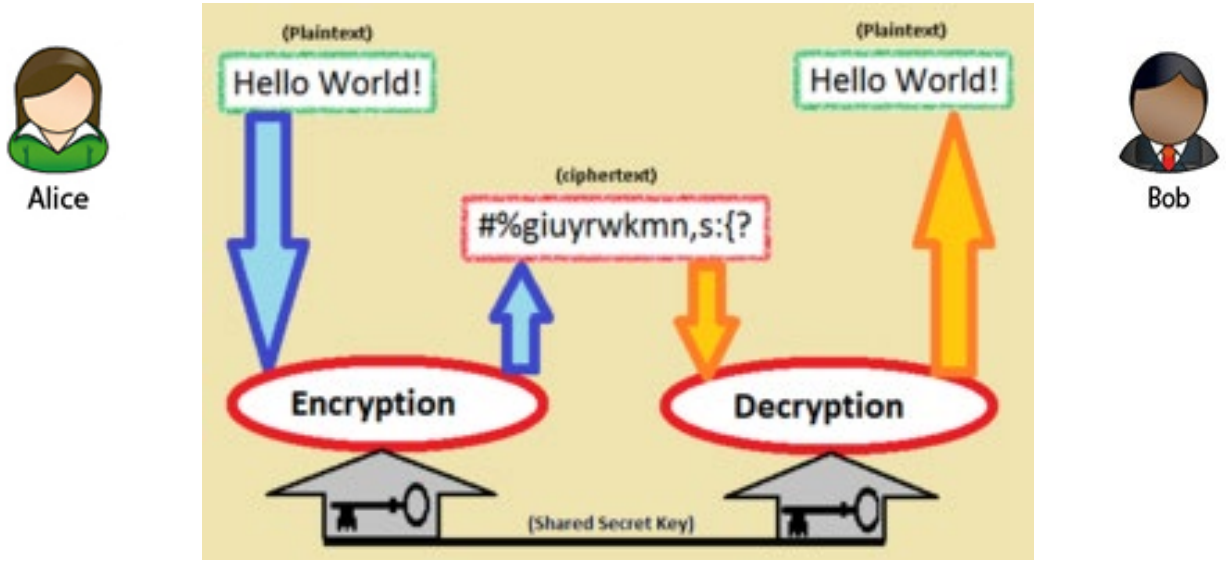
- Explain the difference between classical and modern day cryptography.
- List & describe several representative examples of classical encryption.
- Describe the evolution of symmetric cryptography – from DES to 3DES and AES, and their current day uses.
- Explain the basics of asymmetric cryptography, and current day uses of Diffie Hellman and RSA encryption algorithms.
- Discuss the use of public-key cryptography for purposes of message integrity, authentication & digital signatures.

Required Reading

Computer Security, Stallings: **Chapter 2**
Sections 20.1 to 20.3
Sections 21.1 to 21.5

Introduction

- **Cryptography** – process/technique(s) of converting data into unintelligible form in order to ensure: confidentiality, data integrity, and authentication
  - requirement 1: no data should be lost during encryption
  - requirement 2: decryption should ensure perfect data recovery

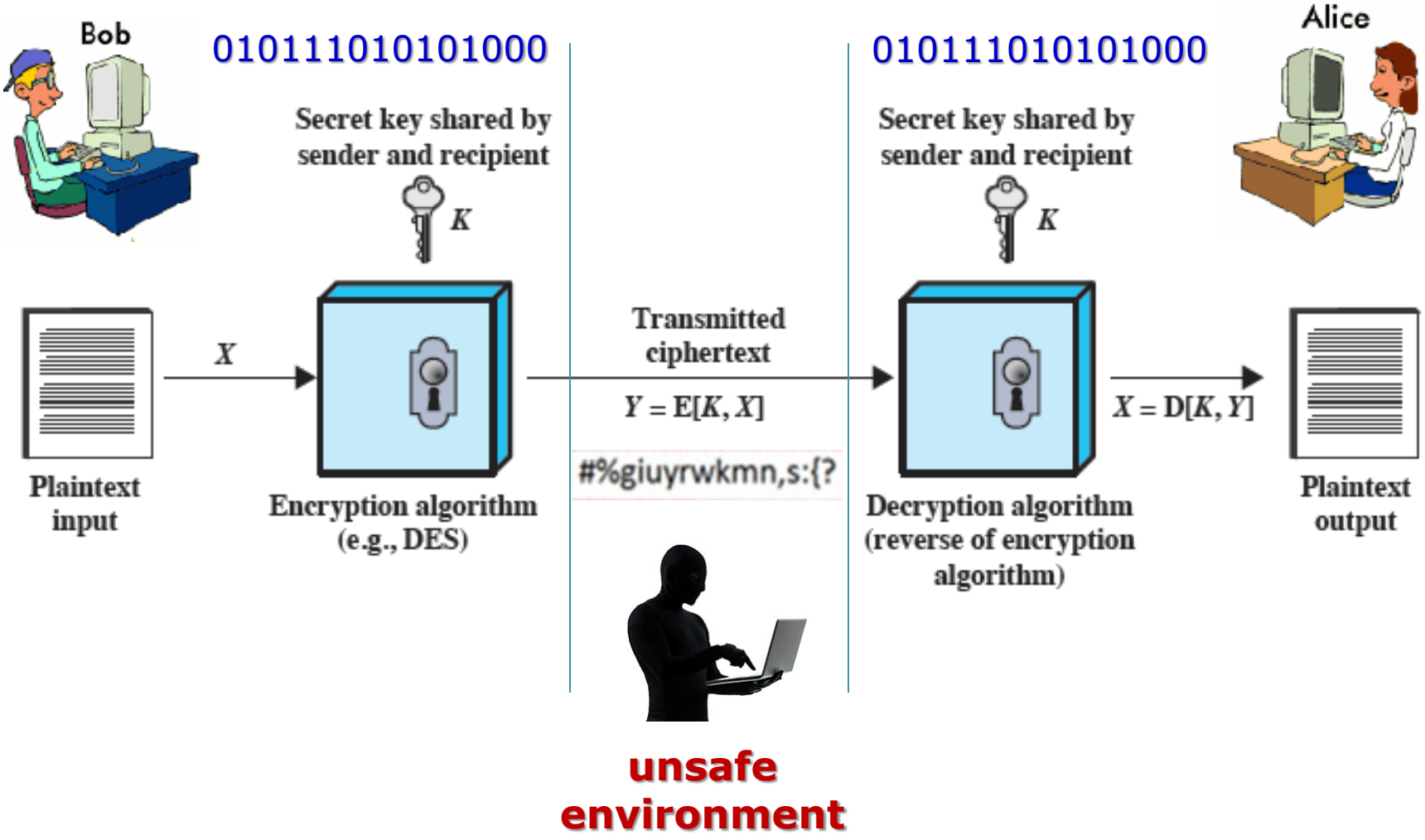


Introduction (cont.)

- **Elements of Encryption System**
  - **plaintext** – original message that should be 'protected'
  - **encryption algorithm** – performs various substitutions, permutations and transformations on plaintext
  - **key** – variable data that is input into encryption algorithm together with plaintext
    - determines exact substitutions, permutations and transformations performed on plaintext
  - **ciphertext** – scrambled message produced as output
  - **decryption algorithm** – encryption algorithm run in reverse

Introduction (cont.)

- **Elements of Encryption System (cont.)**



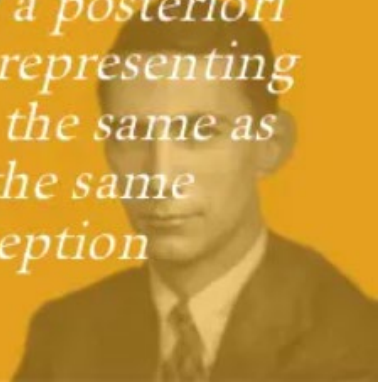
Kerckhoffs' principle

The design of a system should not require secrecy

and compromise of the system should not inconvenience the correspondents

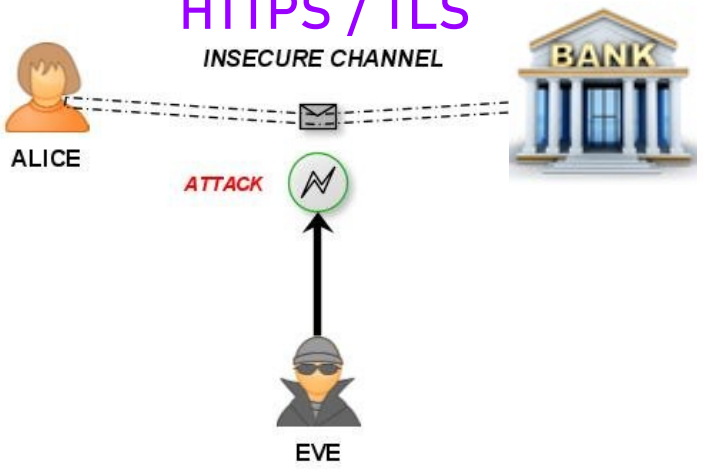
Claude Shannon

"Perfect Secrecy" is defined by requiring of a system that after a cryptogram is intercepted by the enemy, the a posteriori probabilities of this cryptogram representing various messages be identically the same as the a priori probabilities of the same messages before the interception



https://www.slideshare.net/zittersteyn/information-security-paylogic-techtalk-2014/17-Kerckhoffs_principle_The_design_of

Introduction (cont.)

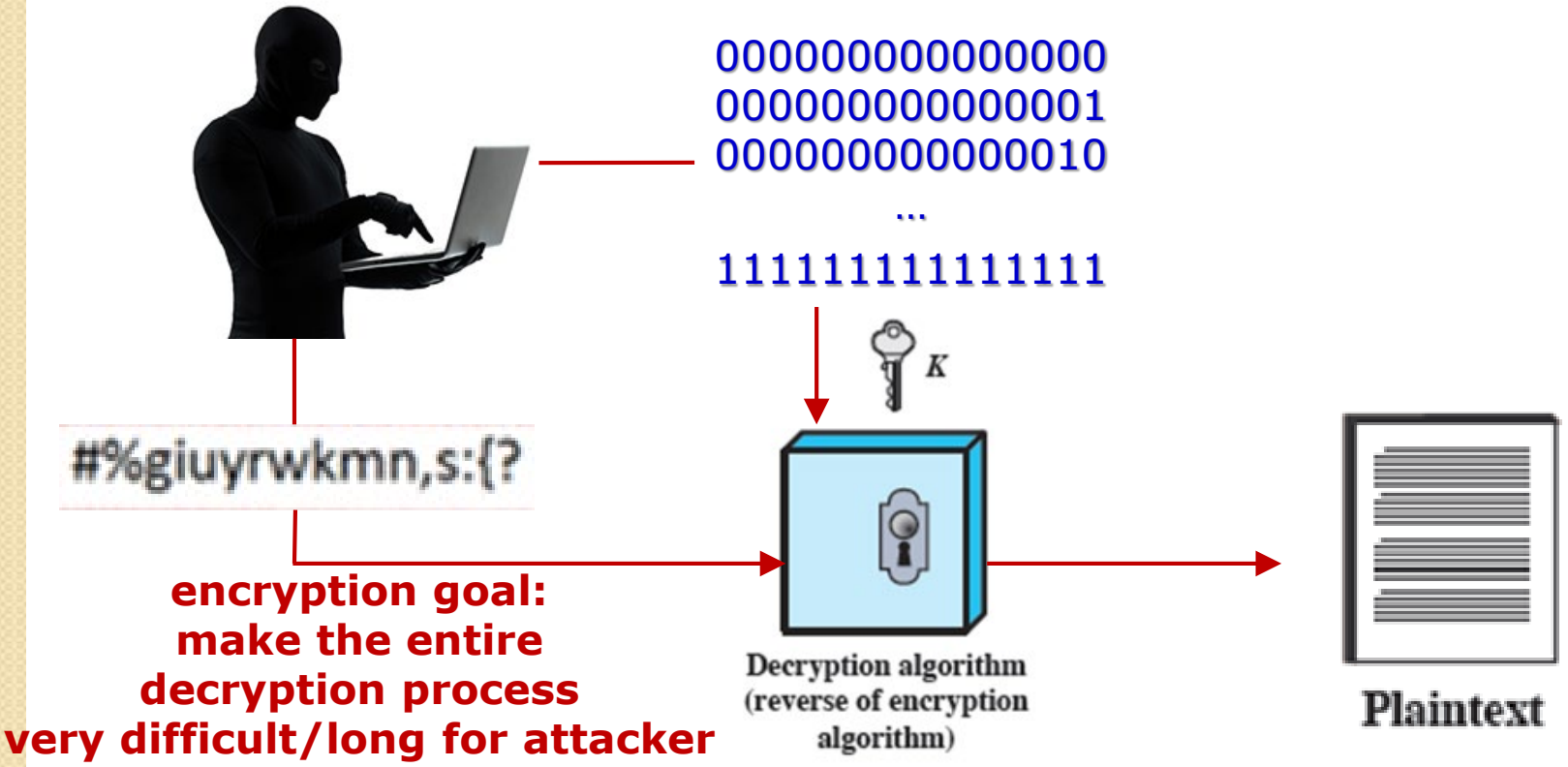



```
Handshake Type: Client Hello (1)
Length: 311
Version: TLS 1.2 (0x0303)
Random
Session ID Length: 0
Cipher Suites Length: 158
Cipher Suites (79 suites)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384 (0xc030)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384 (0xc02c)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 (0xc028)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 (0xc024)
    Cipher Suite: TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA (0xc014)
    Cipher Suite: TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA (0xc00a)
    Cipher Suite: TLS_SRP_SHA_DSS_WITH_AES_256_CBC_SHA (0xc022)
```

```
Handshake Type: Server Hello (2)
Length: 57
Version: TLS 1.2 (0x0303)
Random
Session ID Length: 0
Cipher Suite: TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 (0xc02f)
Compression Method: null (0)
```

Introduction (cont.)

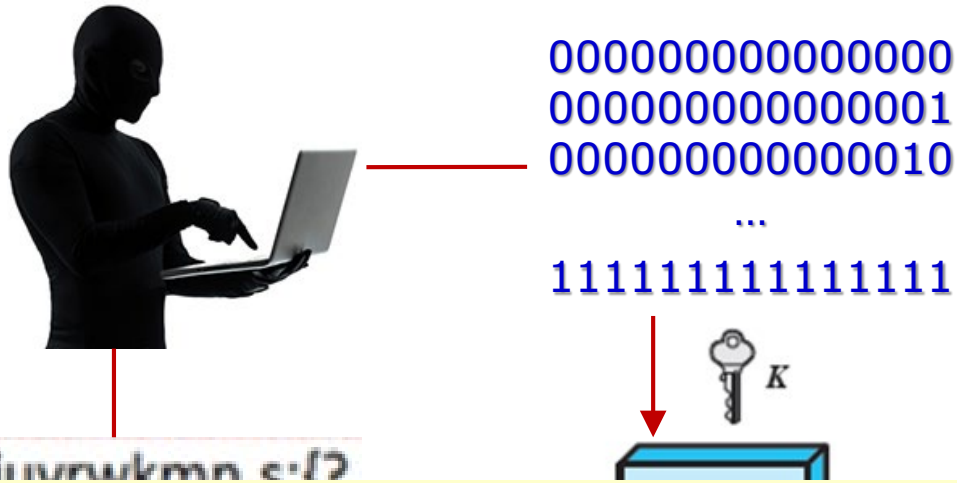
- **Process of Breaking a Cipher**
  - in modern cryptography encryption/decryption algorithm is not a secret
  - hacker probes various keys on the captured ciphertext



Introduction (cont.)

- **Process of Breaking a Cipher**

Assume a hacker does not know the key.
Can he still 'decrypt' a ciphertext?



If the key-size is N [bits],
how big is the key 'space'?

$n_{keys} = 2^N$

decryption process
very difficult/long for attacker

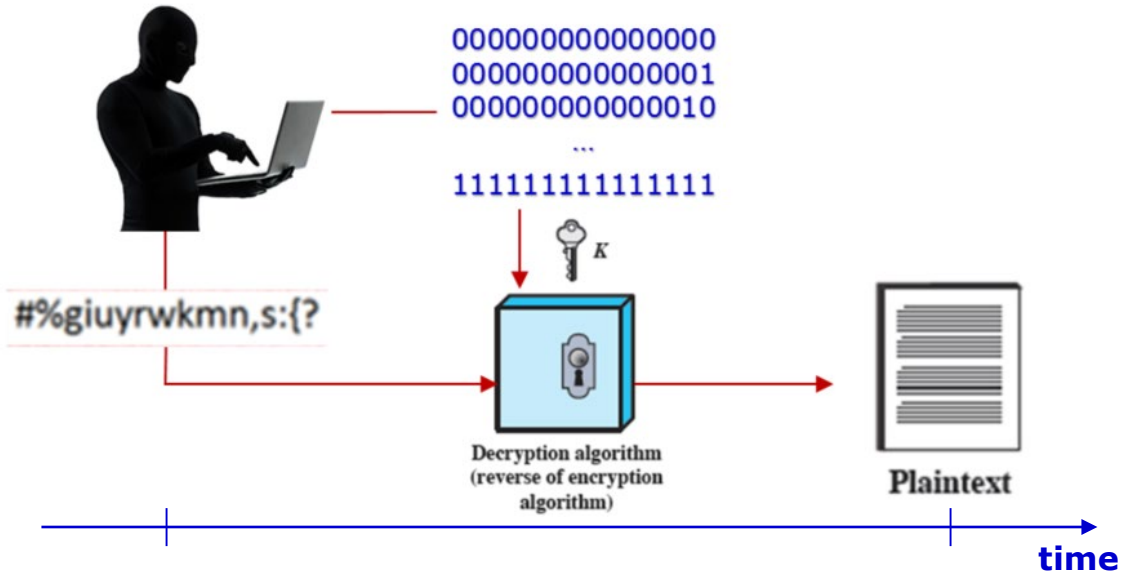
Introduction (cont.)

- **Factors that Impact Success/Speed of Crypto-Attack**
  - number of keys to **try** – n_{keys}
  - time to perform one decryption – $t_{one\text{-}decryption}$

$$\text{crypto-attack speed} = n_{keys} \times t_{one\text{-}decryption}$$

number of **tried** keys

depends on processor speed of attacker's machine



Introduction (cont.)

crypto-attack speed = n_{keys} × $t_{one\text{-}decryption}$

BEST case for hacker:
$n_{keys} = 1$

WORST case for hacker:
$n_{keys} = 2^N$

N bits long keys

000000000000000
000000000000001
000000000000010
…
111111111111111

Introduction (cont.)

- Factors that Influence Success of Crypto-Attack (cont.)
  - **brute force attack on ciphertext** – all possible keys are tried until an intelligible translation into plaintext is obtained
  - with current processing capabilities, 56 bit keys are not considered safe

N bits $n_{keys} = 2^N$ $t_{one\text{-}decrypt} = 1\ [10^{-6}\ sec]$ $t_{one\text{-}decrypt} = 1\ [10^{-12}\ sec]$

Key Size (bits)	Number of Alternative Keys	Time Required at 1 Decryption/μs	Time Required at 10^6 Decryptions/μs
32	$2^{32} = 4.3 \times 10^9$	2^{31} μs = 35.8 minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	2^{55} μs = 1142 years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	2^{127} μs = 5.4×10^{24} years	5.4×10^{18} years
168	$2^{168} = 3.7 \times 10^{50}$	2^{167} μs = 5.9×10^{36} years	5.9×10^{30} years
26 characters (permutation)	$26! = 4 \times 10^{26}$	2×10^{26} μs = 6.4×10^{12} years	6.4×10^6 years

Introduction (cont.)

numbers historically increased

numbers historically decreased

$$\text{crypto-attack speed} = n_{keys} \times t_{one\text{-}decryption}$$

depends on:

- **processor speed of attacker's machine** – decreases due to Moor's l.
- **complexity of crypto algorithm**

defender's goal is to make this speed/time as long as possible.

This is where the emphasis of modern cryptography is.

How about we use/invent very complex algorithms.
Good or bad idea ??

Introduction (cont.)

- **Cryptography vs. Cryptanalysis**

	Cryptography	Cryptanalysis
Defintion	The art or science of encrypting plain messages into cipher text for security of the messages especially while transmission.	The art of obtaining plain text from a cipher text without knowledge of key.
Origin	From Greek κρυπτός, "hidden, secret"; and γράφειν, graphein, "writing", or -λογία, -logia, "study", respectively	From Greek kryptós, "hidden", and analýein, "to loosen" or "to untie"
Practitioner	Cryptographer	Cryptanalyst
Focus	Secret writing	Breaking secrets

http://www.differencebetween.info/difference-between-cryptography-and-cryptanalysis

Introduction (cont.)

The Cryptographer's Dilemma

As with many analysis techniques, having very little ciphertext inhibits the effectiveness of a technique being used to break an encryption. A cryptanalyst works by finding patterns. Short messages give the cryptanalyst little to work with, so short messages are fairly secure with even simple encryption.

Substitutions highlight the cryptologist's dilemma: An encryption algorithm must be regular for it to be algorithmic and for cryptographers to be able to remember it. Unfortunately, the regularity gives clues to the cryptanalyst.

There is no solution to this dilemma. In fact, cryptography and cryptanalysis at times seem together like a dog chasing its tail. First, the cryptographer invents a new encryption algorithm to protect a message. Then, the cryptanalyst studies the algorithm, finding its patterns and weaknesses. The cryptographer then sets out to try to secure messages by inventing a new algorithm, and then the cryptanalyst has a go at it. It is here that the principle of timeliness from Chapter 1 applies; a security measure must be strong enough to keep out the attacker only for the life of the data. Data with a short time value can be protected with simple measures.

Introduction (cont.)

Every Cryptographer Has to Be a Good Cryptanalyst

Every cryptographer's aim is naturally to design an algorithm that won't supply any practically usable results when cryptanalyzed. This doesn't necessarily mean that it can't be cryptanalyzed at all. It normally means that it would take too long (the encrypted information might become worthless in the meantime), or that it would be too costly to justify the value of the information.

For instance, the encryption methods used at the fronts in World War I had been estimated by the cryptologists to require at least one day's work for the adversary to recover the plaintext. After one day, the encrypted commands had become worthless—the shells had long hit by that time. The catch in the matter could only have been that the adversary deciphered faster than expected [BauerMM].

Cryptology Unlocked
By Reinhard Wobst

Introduction (cont.)

Example: Is the best encryption always necessary?

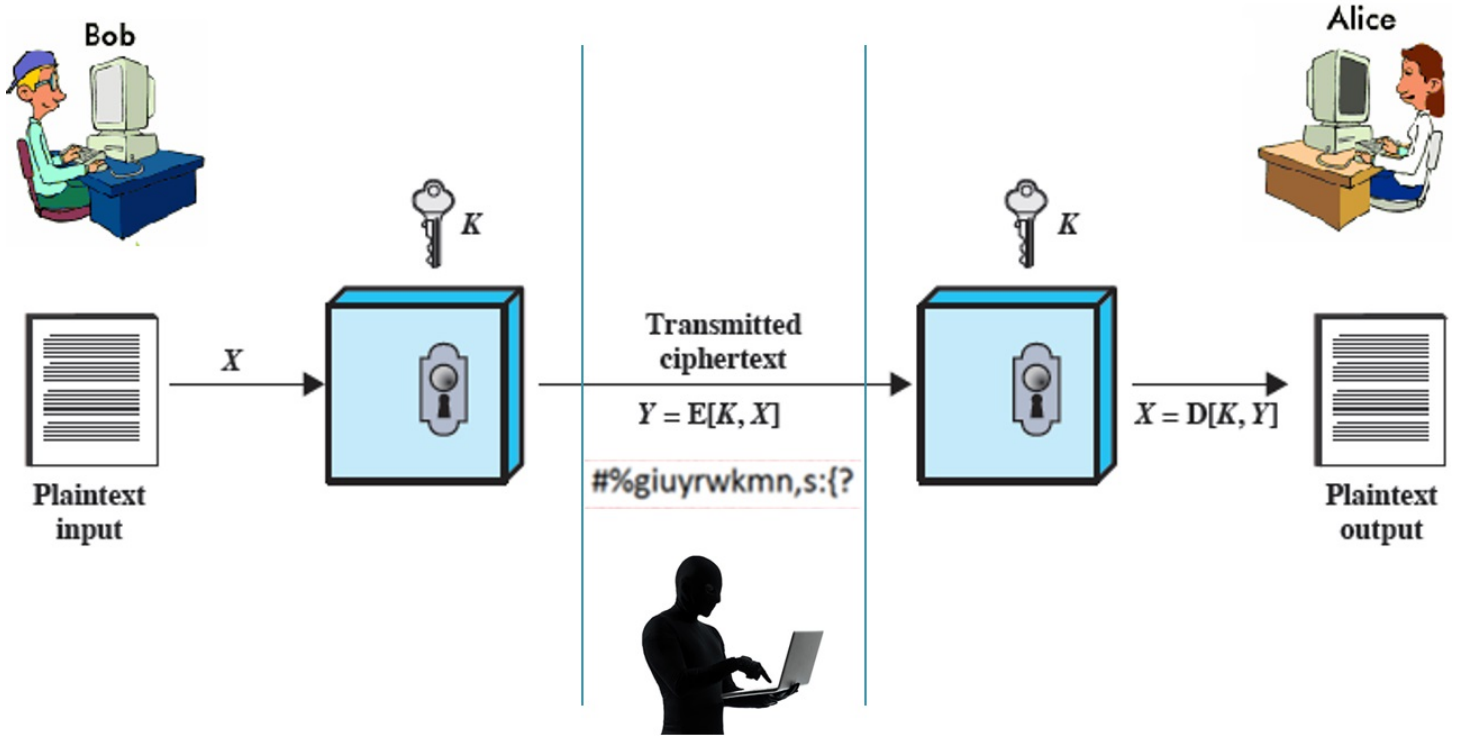
#%giuyrwkmn,s:{?

data(time$_1$) valid for only Δt seconds

Encryption that keep intruder 'busy' for > Δt seconds may be good enough!

Introduction (cont.)

- **Cryptanalysis Attack Models**
  - describe different possible attack scenarios – i.e., type of access a cryptanalyst (hacker) has to a system under attack when attempting to 'break' ciphertext



Introduction (cont.)

- **Examples of Cryptanalysis Attack Models**

Ciphertext-only attack: The key or plaintext is revealed exclusively by means of the ciphertext. This method is the most difficult. If too little is known of the rules of the ciphertext to be able to exploit them, only one obvious thing remains: trying every possible key. This is called **brute-force attack** (exploiting the key space; exhaustion method). Often, however, it is sufficient to try just a few keys; but more about this later.

Known-plaintext attack: Part of the plaintext is known in addition to the ciphertext, and used to reveal the remaining plaintext, normally by means of the key. This is perhaps the most important cryptanalytic method, because it is much more powerful than a ciphertext-only attack and normally possible: the attacker guesses certain words in the text; the beginning of the text is fixed; known, uncritical plaintexts are encoded with the same key as confidential plaintexts, etc.

passive attacks

hacker does **NOT** have access to crypto-system

Chosen-plaintext attack: In this attack, the adversary has the ability to obtain the encryption of any plaintext(s) of its choice. It then attempts to determine the plaintext that was encrypted to give some other ciphertext.

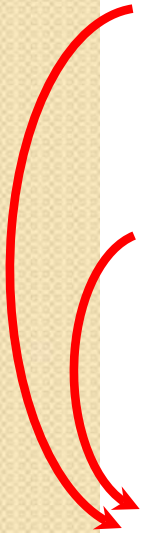
Chosen-ciphertext attack: The final type of attack is one where the adversary is even given the capability to obtain the decryption of any ciphertext(s) of its choice. The adversary's aim, once again, is then to determine the plaintext that was encrypted to give some other ciphertext (whose decryption the adversary is unable to obtain directly).

active attacks

hacker has access to crypto-system

https://www.u-cursos.cl/usuario/777719ab2ddbbdb16d99df29431d3036/mi_blog/r/1_book-introduction_to_modern_cryptography.pdf

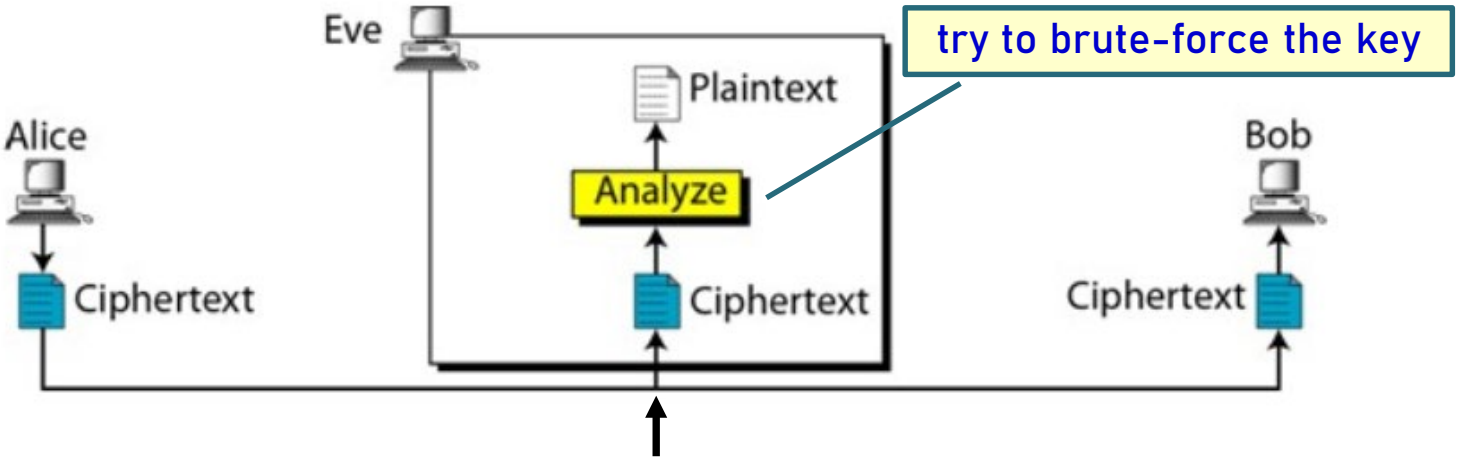
Type of Attack	Known to Cryptanalyst
Ciphertext Only = brute force	• Encryption algorithm ← known • Ciphertext ← obtained
Known Plaintext [accelerated cryptanalysis possible]	• Encryption algorithm • Ciphertext • One or more plaintext-ciphertext pairs formed with the secret key ← can be guessed
Chosen Plaintext [attacker has access to the system with secret key setup (as black box) & can enter chosen plaintexts]	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen Ciphertext [attacker has access to the system with secret key setup (as black box) & can enter chosen ciphertexts]	• Encryption algorithm • Ciphertext • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen Text [attacker has access to the system with secret key setup (as black box) & can enter both, chosen plaintexts and ciphertexts]	• Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key



Introduction (cont.)

Ciphertext Only Attacks: goal is to find the plaintext

(nothing about plaintext is known !!!)

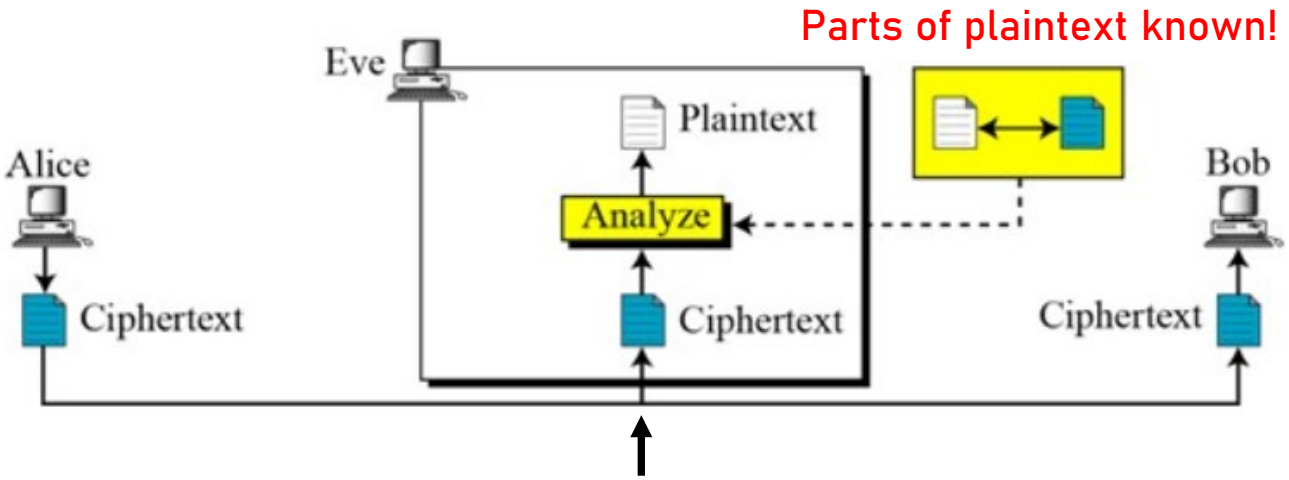



```
8DGcvHRafNTgTuyYe2vFPepZlBUrswPGTmPP1cnw3ZBxpHB3be
PpxFt+8X8bmdMpxwha6So6C352DeZE93dFbUpTk8aTfyYESNh+
aYPfVEYl/6+1a2gpJ7Rdj1oCRtHy/Il6RechMjsl+wrjNHRWr+
yP0NncBPShXrG+vEraQlgi4BaETLgA2/rtZdWcaJjBGSDRyghE
qLBuiAFAjL1ilJ8pX3SfPYRDVIl4/o2LSZJVCMlVpauz5mX5Yf
IUZCzqD2RfmPpcW/un6NhO5oLZIB/9WYAMrHVCSXwkxzwOau6s
7IhubbU16QWAqF2lGkR1yBjz9PO8L19MYFYrjxlj1MOYtvrs5V
wXBEGzpu6xDlOP344uR9cyOW8gY7JLG207a1lNqtrF4dLD6ZaS
Pywo2VRcDr+kTRLv/3TKWPY1bpG3qG0l9fSBo2lFPjxTPSnFEt
/ZfU3V1w6y7KMyWxcySHXzX2PwC9Wj6DluiijDfikoXjR4Lqtu
griG+TprpPWmpCST9LwvRoe14Yh6EnYCTvYUyvZVy1foS4xdO3
CtgcYDOmZm2Vzlp+s27s9zfdGwe5YV41+ucLHCf+6o5rMrN9RV
B/H2K0hpo2bxpgH+/Zef3d3xGqPydA
```

Introduction (cont.)

Known Plaintext Attacks: goal is to find the key and then apply it to the entire ciphertext

(part of plaintext is known!)



Hello Bob,

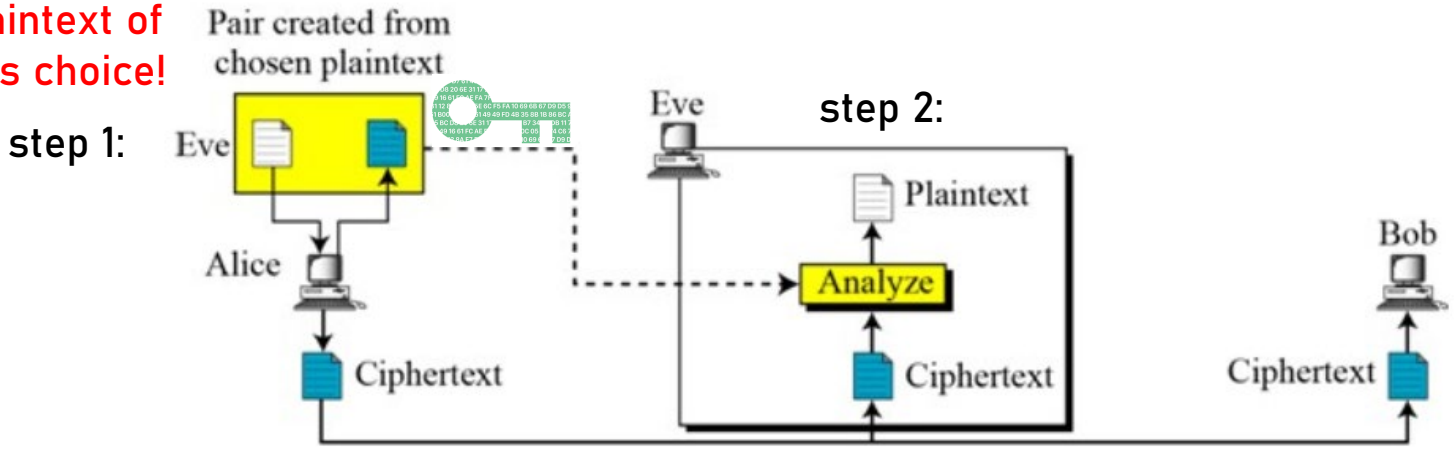
```
Wj6DluiijDfikoXjR4L

0DGcvHRafNTgTuyYe2vFPepZlBUrswPGTmPP1cnw3ZBxpHB3be
PpxFt+8X8bmdMpxwha6So6C352DeZE93dFbUpTk8aTfyYESNh+
aYPfVEYl/6+1a2gpJ7Rdj1oCRtHy/Il6RechMjsl+wrjNHRWr+
yP0NncBPShXrG+vEraQlgi4BaETLgA2/rtZdWcaJjBGSORyghE
qLBuiAFAjL1ilJ8pX3SfPYRDVIl4/o2LSZJVCMlVpauz5mX5Yf
IUZCzqD2RfmPpcW/un6Nh05oLZIB/9WYAMrHVCSXwkxzwOau6s
7IhubbU16QWAqF2lGkR1y8jz9PO8L19MYFYrjxlj1MOYtvrs5V
wXBEGzpu6xDlOP344uR9cyOW8gY7JLG207a1lNqtrF4dLD6ZaS
Pywo2VRcDr+kTRLv/3TKWPY1bpG3qG0l9fSBo2lFPjxTPSnFEt
/ZfU3V1w6y7KMyWxcySHXzX2PwC9Wj6DluiijDfikoXjR4Lqtu
griG+TprpPWmpCST9LwvRoe14Yh6EnYCTvYUyvZVy1foS4xd03
CtgcYDOmZm2Vzlp+s27s9zfdGwe5YV41+ucLHCf+6o5nMnN9RV
```

Introduction (cont.)

lunchtime attacks ☺

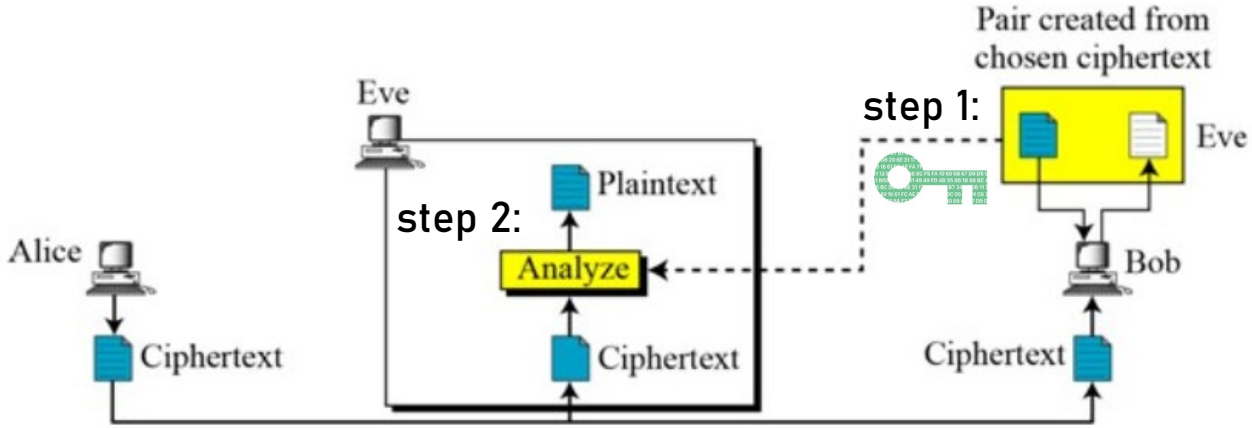
Chosen Plaintext Attacks: goal is to find the key

Any plaintext of hacker's choice!



Chosen Ciphertext Attacks: goal is to find the key

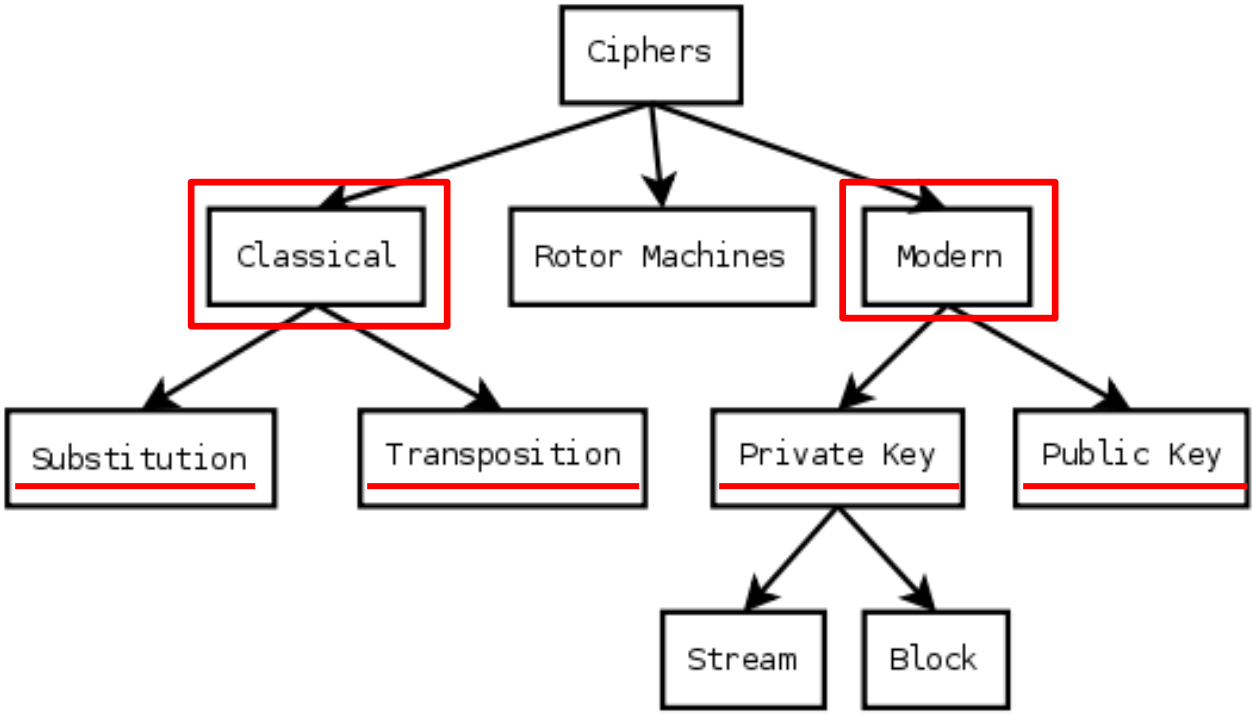
Any ciphertext of hacker's choice!



Eve gets access to the system once, manages to 'crack' the key and then (re)uses this key to decrypt any subsequent messages …

Ciphers

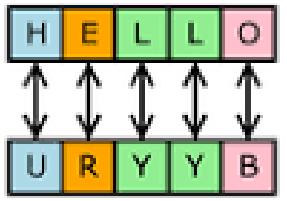
- **History of Cryptography**
  - humans have been using cryptographic techniques for 1000s of years – what have changed are the complexity and creativity of cryptographic techniques



Ciphers (cont.)

- **Classical vs. Modern Cryptography**
  - **Classical cryptography** - more of an art than science
    - schemes were designed in an ad-hoc manner and then evaluated based on their perceived complexity/cleverness
    - true 'strength' of these schemes was in 'secrecy' of their respective protocols
  - **Modern cryptography** - based on scientific foundations
    - the strength is NOT in secrecy of protocols but in sound mathematical and computational principles
    - it is now possible to formally argue about the security protocols
    - used for more than just data confidentiality - can protect data integrity, enable user authentication, etc.

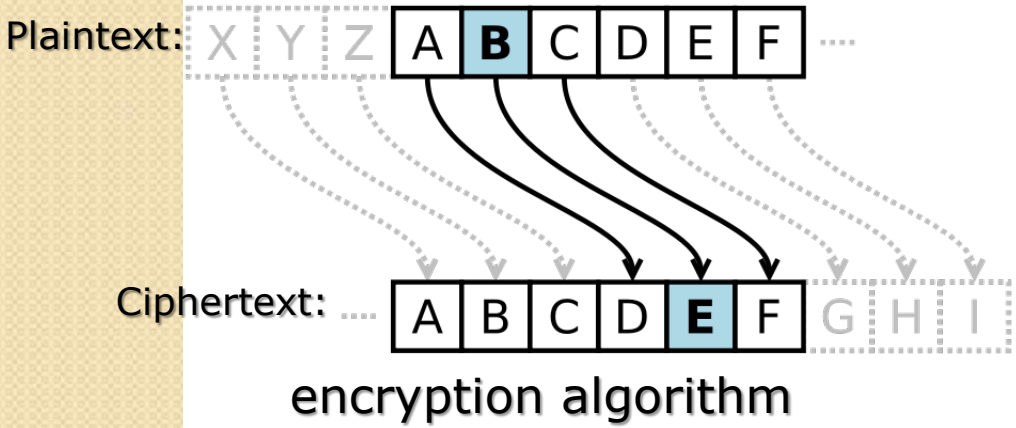
Classical Ciphers

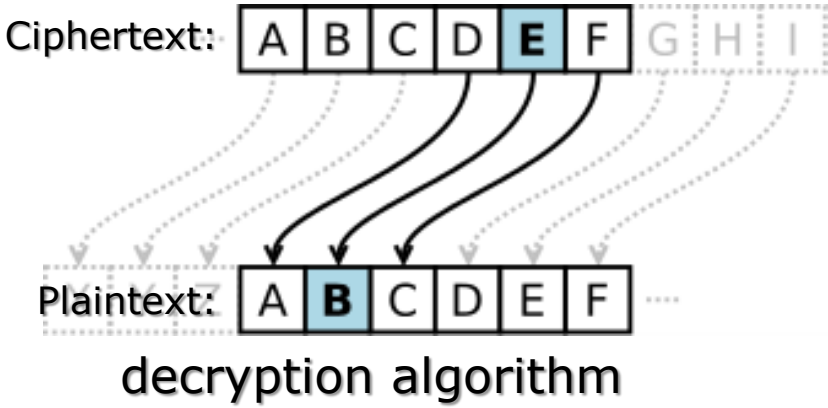


- **Substitution Cipher** – the units of plaintext (letters) are kept in the same original sequence, but the units themselves are altered

- **Caesar Cipher** – monoalphabetic substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the alphabet

Example: Caesar Cipher with **k=3**





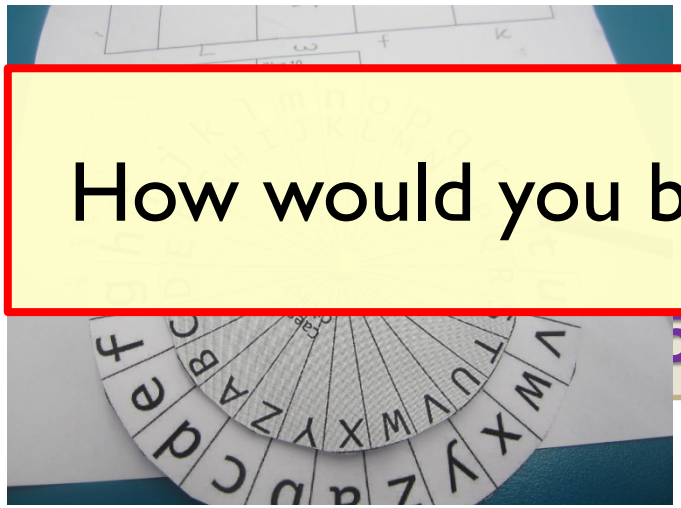
Classical Ciphers (cont.)

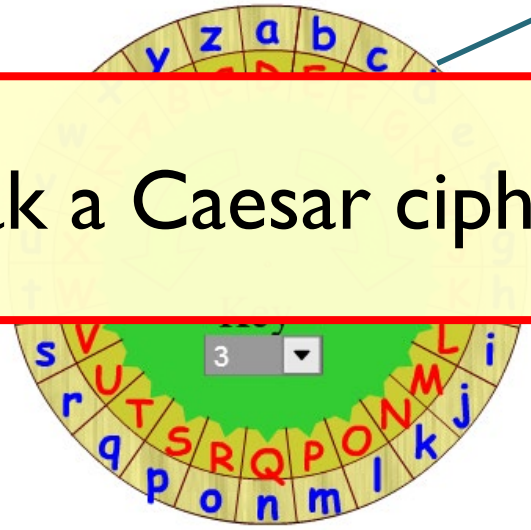
Example: Caesar cipher encryption with k=3

Ciphertext: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

Plaintext: THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG

https://www.cs.uri.edu/cryptography/classicalshiftdemo.htm
http://www.simonsingh.net/The_Black_Chamber/caesar.html

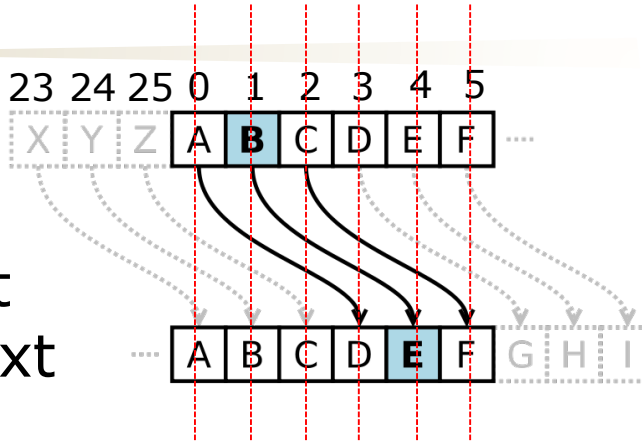




How would you break a Caesar cipher?

Classical Ciphers (cont.)

Represent letters with numbers!



- **Cesar Cipher as an Algorithm**

T_i - *i*-th character of the plain text
C_i - *i*-th character of the cipher text
i = 0, 1, 2, .. , m-1 in English
m - length of the alphabet
k - shift

Encryption: $\mathbf{C_i = (T_i + k) \bmod m}$

Decryption: $\mathbf{T_i = (C_i - k) \bmod m}$

NOTE:

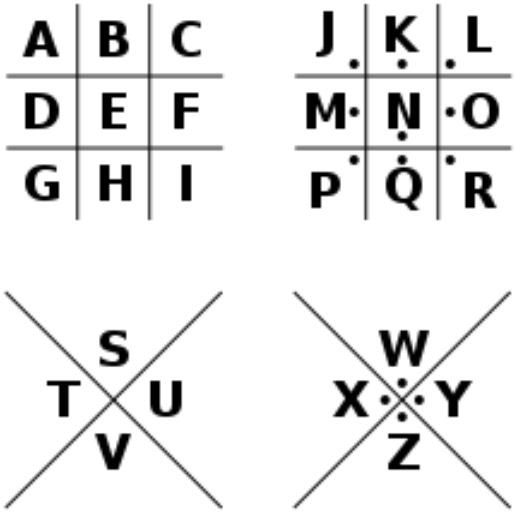
-b mod m = (-b + m) mod m

http://en.algoritmy.net/article/40203/Caesar-cipher

Classical Ciphers (cont.)

aka masonic or tic-tac-toe cipher

- **Pigpen Cipher** – simple substitution cipher in which each letter is replaced with a graphical symbol
  - alphabet is written in 4 grids shown below
  - each letter is replaced with a symbol that corresponds to the portion of the pigpen grid that contains the letter
  - used by Freemasons in the 18th Century to keep their records private

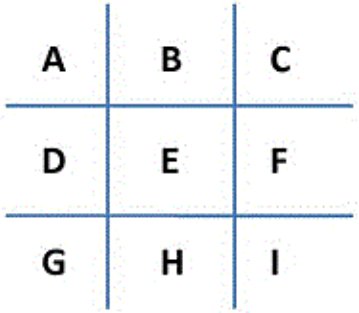


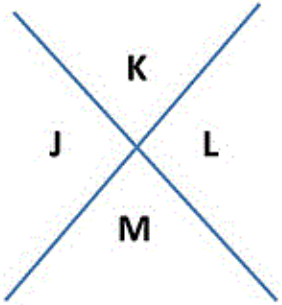
Example: Pigpen cipher

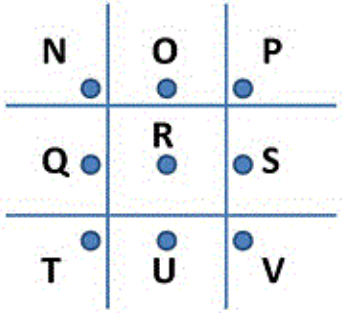
SECRET

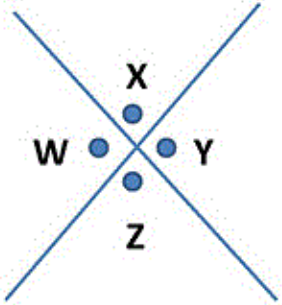
Classical Ciphers (cont.)

Example: Pigpen Cipher variants









https://crypto.interactive-maths.com/pigpen-cipher.html

Classical Ciphers (cont.)

Example: How to break a simple substitution cipher FASTER !?

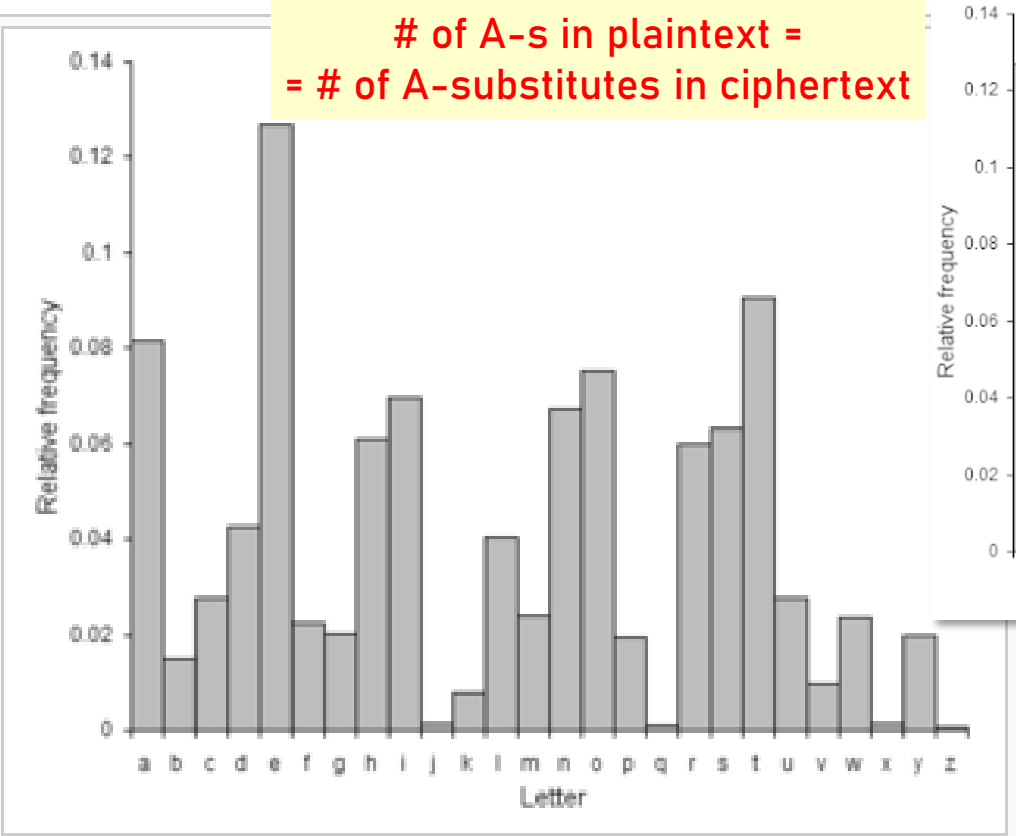
"Zahyz tlhu kpmmlylua aopunz av kpmmlylua wlvwsl. Mvy ayhclsslyz, zahyz alss aolt ddlyl aolf hyl, ddlyl aolf hyl nvpun. Mvy vaolyz, aolf hyl qbza spaasl spnoaz pu aol zrf. Mvy zjovshyz, aolf hyl aol dvysk vm aol buruvdu, fla av il kpzjvclylk huk buklyzavvk. Mvy tf ibzpulzzthu, aolf hyl nvsk. Iba hss zahyz zahf zpslua. Huk fvb? Uv vul lszl pu aol dvysk dpss zll aol zahyz hz fvb kv… Mvy fvb, huk vusf mvy fvb, aol zahyz dpss hsdhfz il shbnopun."

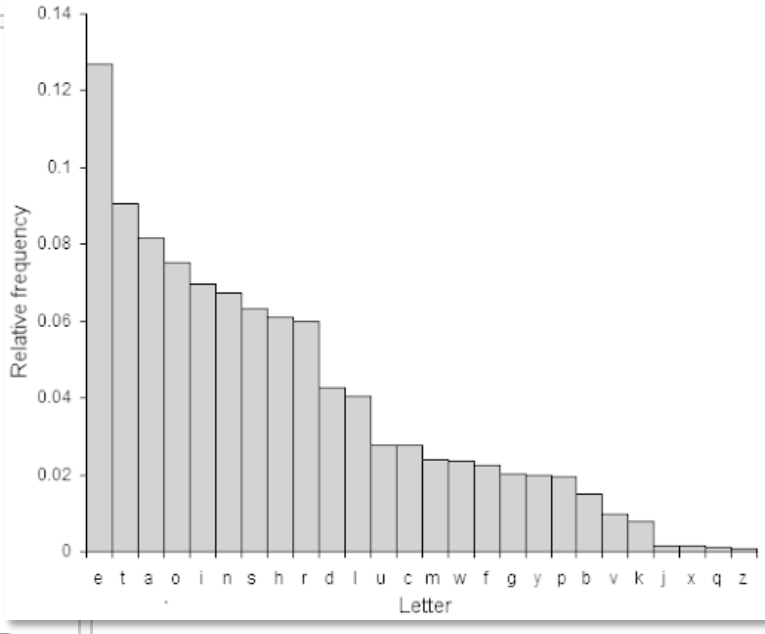
"Stars mean different things to different people. For travellers, stars tell them where they are, where they are going. For others, they are just little lights in the sky. For scholars, they are the world of the unknown, yet to be discovered and understood. For my businessman, they are gold. But all stars stay silent. And you? No one else in the world will see the stars as you do… For you, and only for you, the stars will always be laughing."

Classical Ciphers (cont.)

Example: How to break a simple substitution cipher?

of A-s in plaintext =
= # of A-substitutes in ciphertext

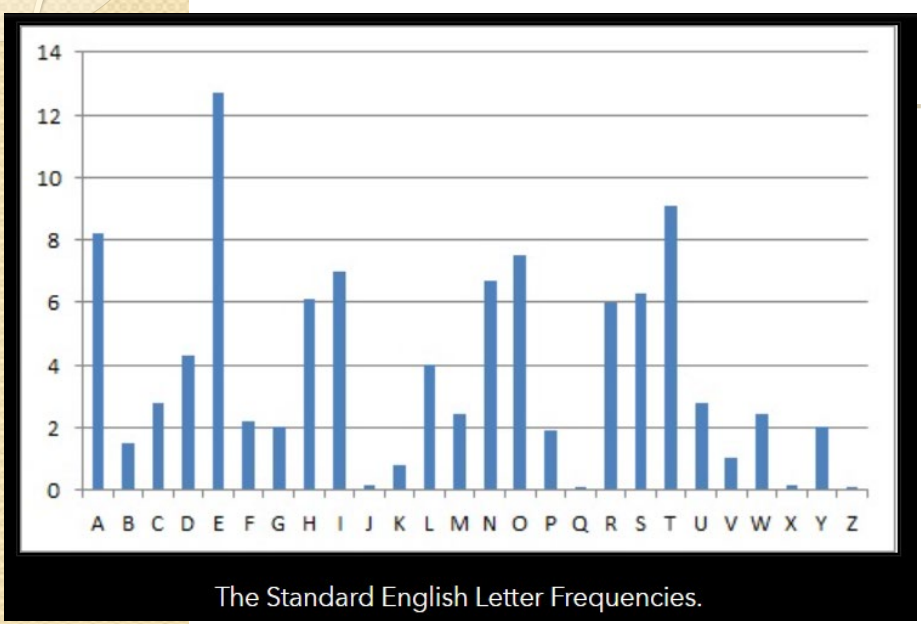




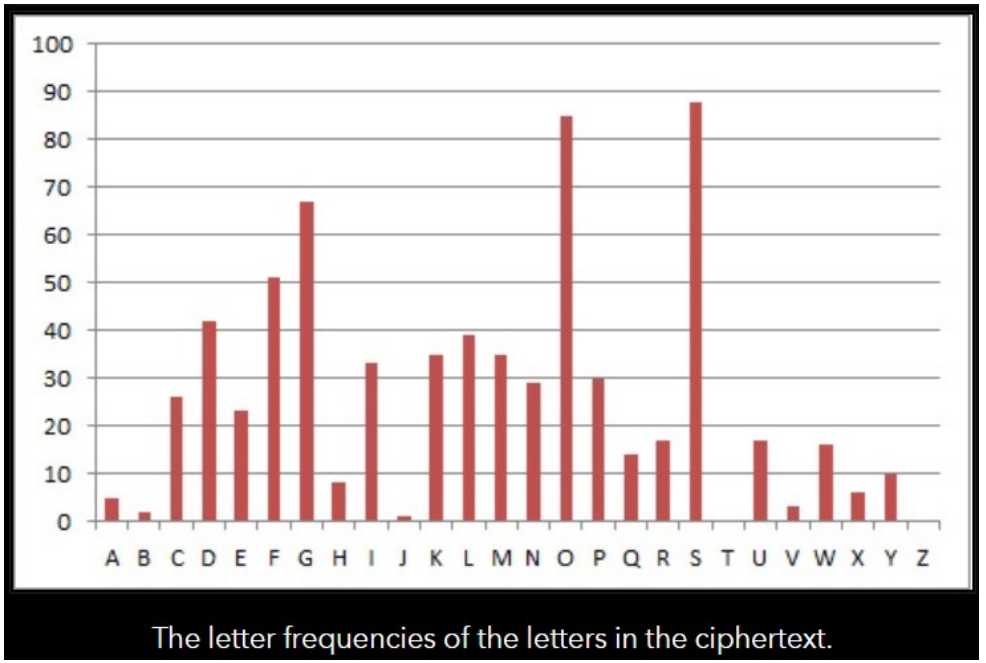
The distribution of letters in a typical sample of English language text has a distinctive and predictable shape. A Caesar shift "rotates" this distribution, and it is possible to determine the shift by examining the resultant frequency graph.

http://www.usafa.edu/df/dfe/dfer/centers/accr/tools/ShiftCipherApplet.html

Example: breaking a cipher using FREQUENCY analysis



The Standard English Letter Frequencies.



The letter frequencies of the letters in the ciphertext.

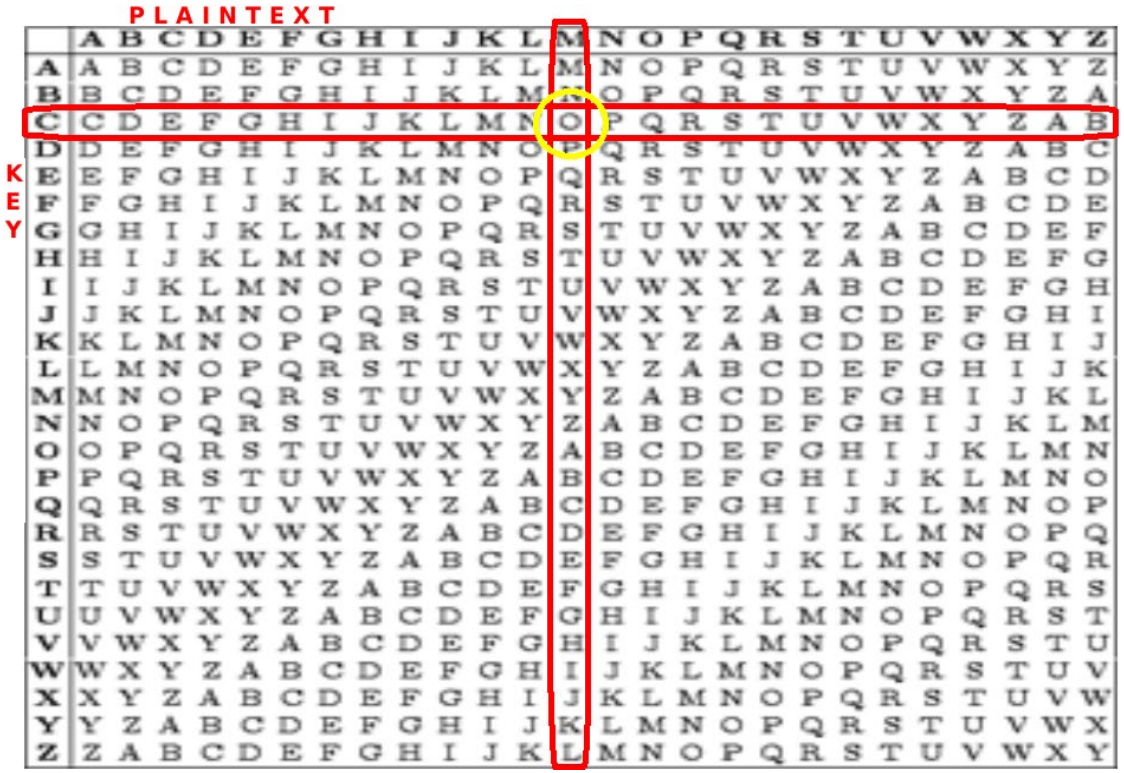
"Now that we have all the frequencies of ciphertext letters, we can start to make some substitutions. We see that the most common ciphertext letter is "S", closely followed by "O". From the chart and table above, we can guess that these two letters represent "e" and "t" respectively, and after making these substitutions we get … "

https://crypto.interactive-maths.com/frequency-analysis-breaking-the-code.html

Classical Ciphers (cont.)

- **Polyalphabetic / Vigenere Cipher** – complex substitution cipher - instead of shifting each character by the same number, characters located at different positions are shifted by different numbers – key keeps changing!

- key (word) must be provided
- key is aligned with plaintext – key-letter determines the value of cipher-letter



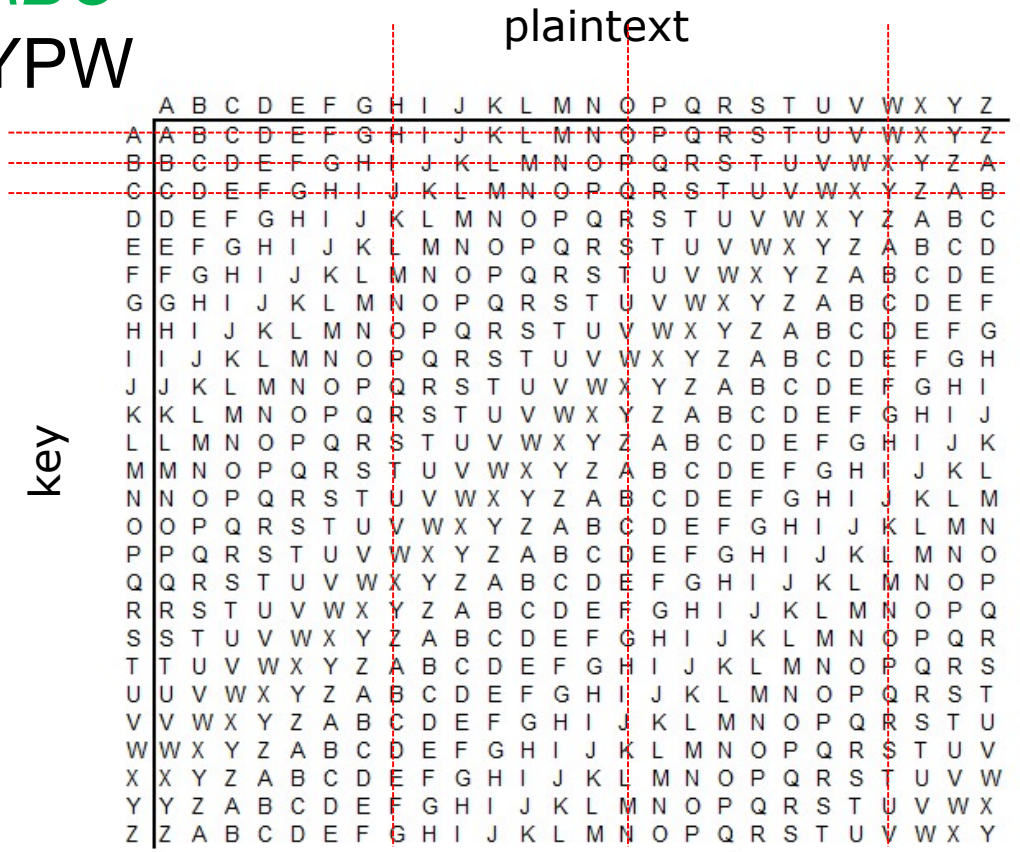
Classical Ciphers (cont.)

Example: Viegenere Cipher - decryption using the table

Plaintext: HOW ARE YOU

Key: *ABC ABC ABC*

Ciphertext: HPY ASG YPW



http://www.simonsingh.net/The_Black_Chamber/vigenere_square_tool.html

Classical Ciphers (cont.)

- **Vigenere Cipher as an Algorithm**

  T_i - *i*-th character of the plain text
  C_i - *i*-th character of the cipher text
  K_i - *i*-th character of the key phrase
  i = 0, 1, 2, .. , m-1
  m - length of the alphabet

  Encryption: $\mathbf{C_i = (T_i + K_i) \bmod m}$

  Decryption: $\mathbf{T_i = (C_i - K_i) \bmod m}$

http://en.algoritmy.net/article/45623/Vigenere-cipher

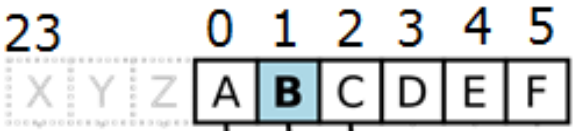
Classical Ciphers (cont.)

Example: Viegenere Cipher - encryption using algorithm

Open text: ATTACK AT DAWN

Key phrase: CAT

Length of the alphabet: 26




```
A + C = 0  + 2  = 2  = C
T + A = 19 + 0  = 19 = T
T + T = 19 + 19 = 12 = M
A + C = 0  + 2  = 2  = C
C + A = 2  + 0  = 2  = C
K + T = 10 + 19 = 3  = D
A + C = 0  + 2  = 2  = C
T + A = 19 + 0  = 19 = T
D + T = 3  + 19 = 22 = W
A + C = 0  + 2  = 2  = C
W + A = 22 + 0  = 22 = W
N + T = 13 + 19 = 6  = G
```

Classical Ciphers (cont.)

Example: Viegenere Cipher - how to decipher ???

Key is not know, but the keyword size is = n.

Ideally would know the number of charactrs in the keyword. Once the keyword is decrypted, the rest is easy …

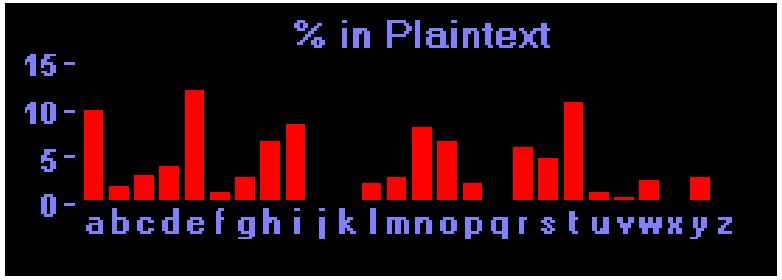
Plaintext: HOW ARE YOU TODAY …
Key: *MUSICMUSICMUSICMUSICMUSIC*
Ciphertext: TIO ITQ SGC VAXSG

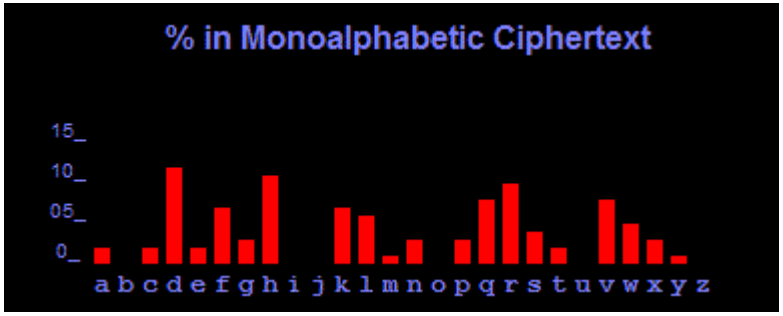
Total number of keys = 26^n.

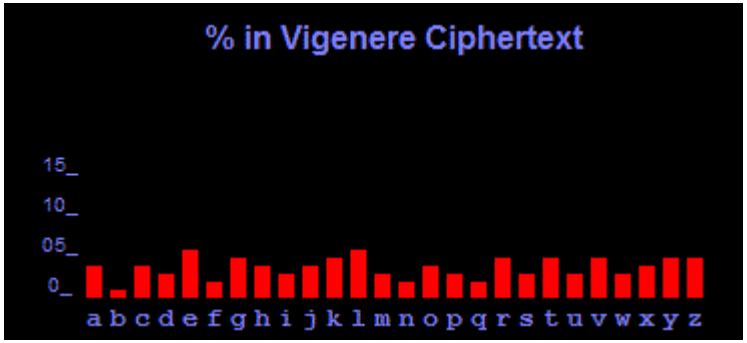
Classical Ciphers (cont.)

- Polyalphabetic / Vigenere Cipher (cont.)
  - Why is it so strong?

Aged twenty six, Vigènere was sent to Rome on a diplomatic mission. It was here that he became acquainted with the writings of Alberti, Trithemius and Porta, and his interest in cryptography was ignited. For many years, cryptography was nothing more than a tool that helped him his diplomatic work, but at the age of thirty nine, Vigènere decided that he had amassed enough money to be able to abandon his career and concentrate on a life of study. It was only then that he began research into a new cipher.





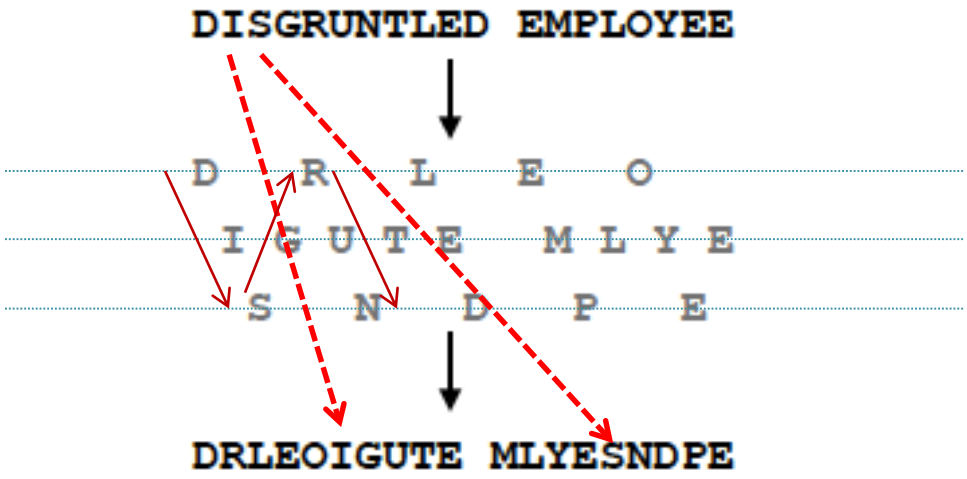


http://www.simonsingh.net/The_Black_Chamber/vigenere_strength.html

Classical Ciphers (cont.)

- **Transposition Cipher** – order of letters in the ciphertext is rearranged according to some predetermined method
- **Rail Fence Cipher** – transposition cipher in which the plaintext is written downwards and upwards on successive *'rails' of an imaginary fence*
  - ➢ the message is then read off in rows

3-rail fence:



Classical Ciphers (cont.)

Example: Rail Fence Cipher

Plaintext: DEFEND THE EAST WALL

2-Rail Fence Ciphertext: DFNTEATALEEDHESWL

D		F		N		T		E		A		T		A		L
	E		E		D		H		E		S		W		L	

3-Rail Fence Ciphertext: DNETLEEDHESWLFTAA

D				N				E				T				L		
	E		E		D		H		E		S		W		L		X	
		F				T				A				A				X

Classical Ciphers (cont.)

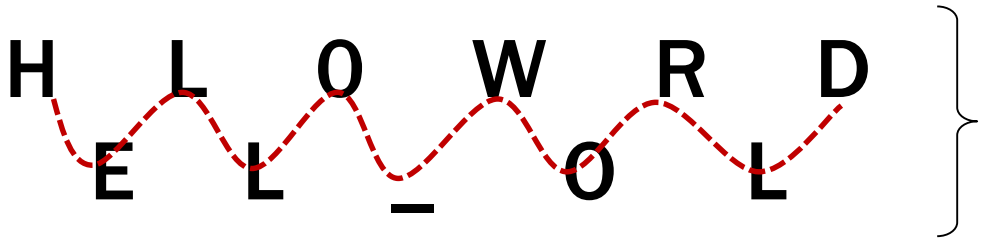
Example: How to break a 2-rail cipher?

HLOWRDEL OL

Decrypting algorithm:

1) Count the letters in the cipher.
2) Divide the letters in 2 equal parts.
3) Draw/write the letters in a 2-rail zigzag pattern with ½ of the letters on the top and ½ of the bottom rail.

If number of letters is odd, add extra letter to the top rail.



HELLO WORLD

Classical Ciphers (cont.)

Example: **How to break a 3-rail cipher?**

M _ AETM _ T6EE _

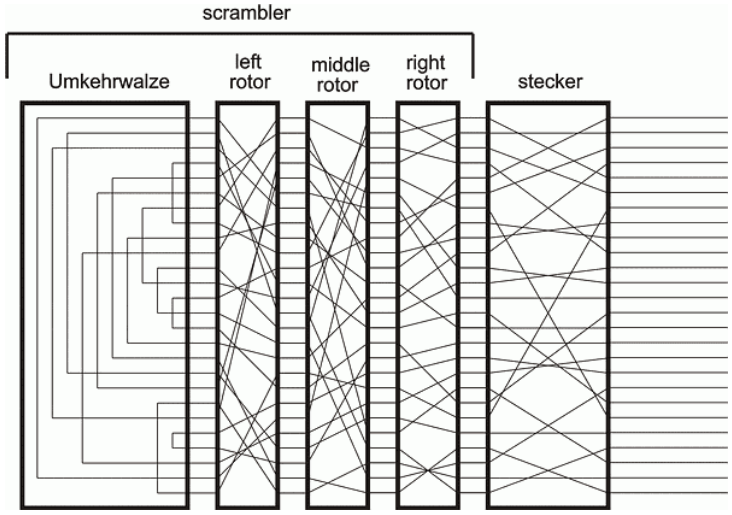
Decrypting algorithm:

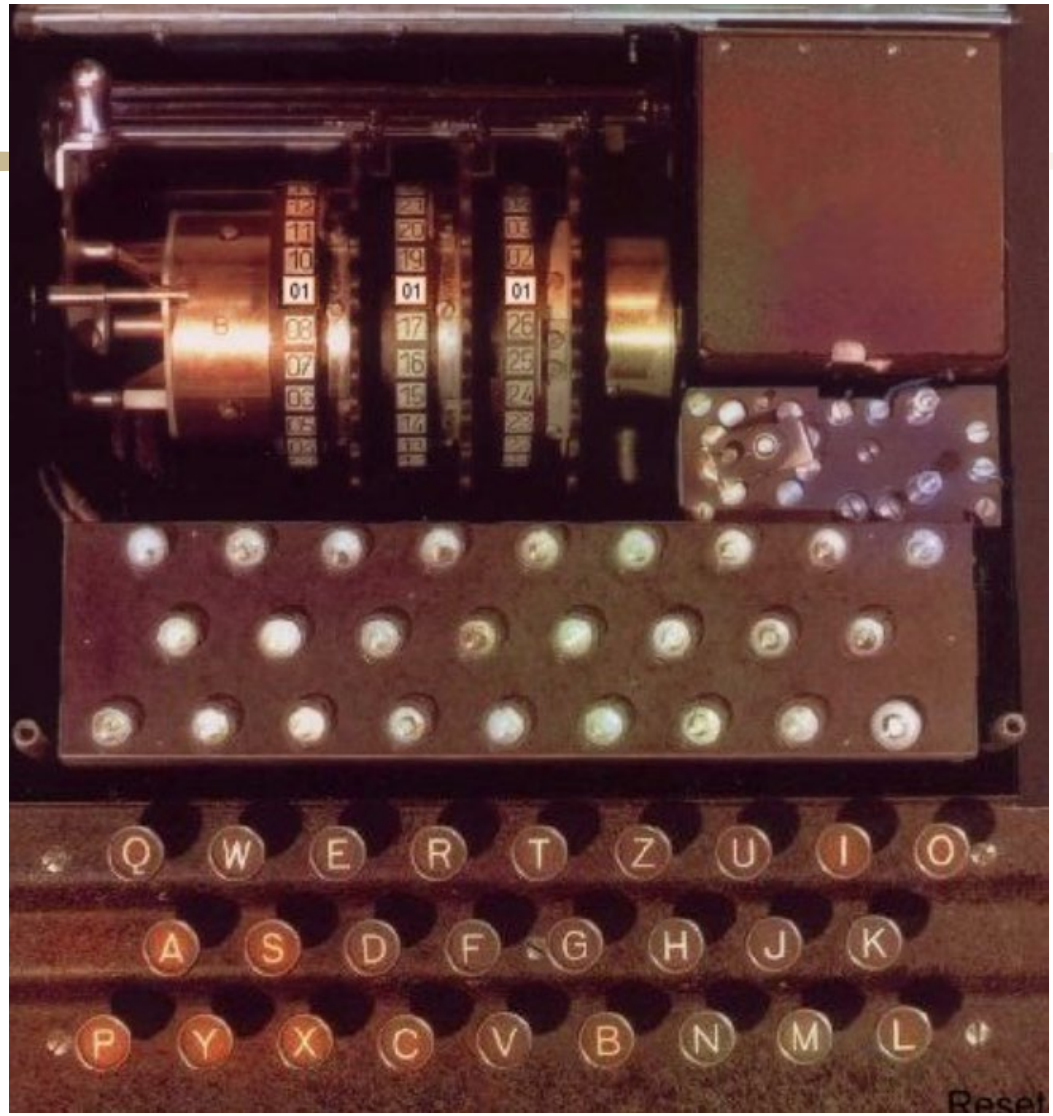
1) Count the number of letters.
2) Make an outline of the zigzag pattern with the given number of rails and given number of letters.
3) Arrange the letters at the allocated spaces …

M				_				A			
	E		T		M		_		T		6
		E				E				_	

Rotor Machines

- **Rotor Machines** – mechanical devices for implementing complex substitution cipher
  - in widespread use 1920 – 1970 – most famous example is German Enigma machine from World War II
  - consists of keyboard (input letter), set of rotors, lights (output letter)
  - every time a key is pressed, some of the rotors change position, producing different output letter





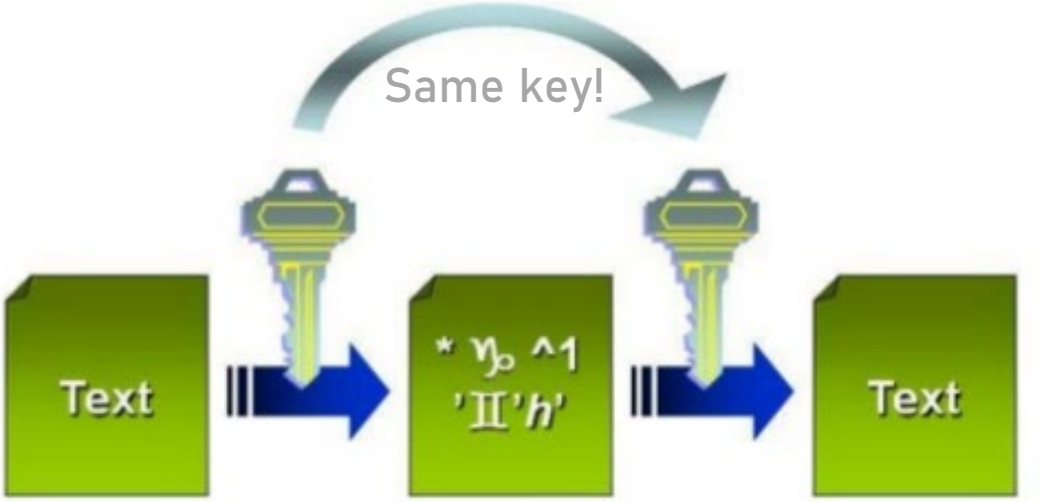
https://www.advanced-ict.info/javascript/enigma.html

Rotor Machines (cont.)

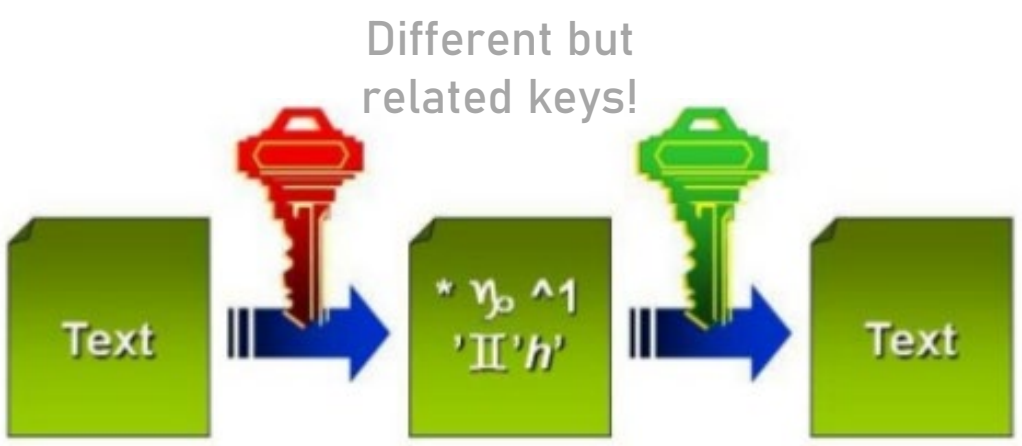
http://www.telegraph.co.uk/culture/film/11229586/Imitation-Game-how-did-the-Enigma-machine-work.html

Modern Cryptography

Symmetric Encryption

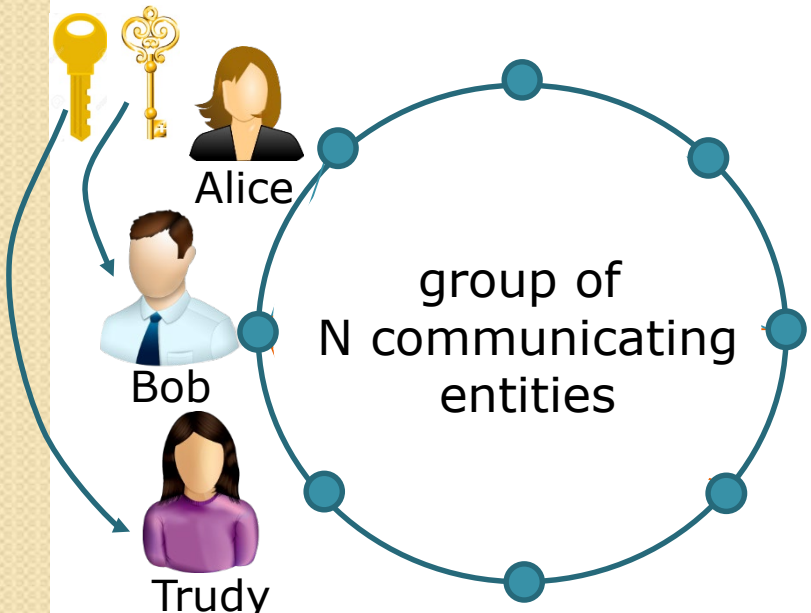


Public Encryption



Symmetric Ciphers

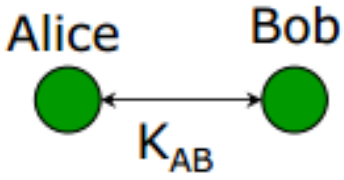
- **Symmetric Encryption** – private-key encryption - uses the same secret/private key to encrypt & decrypt information
  - symmetric key = shared secret – must only be known to the communicating parties – challenge # 1
  - to ensure full confidentiality in a group of N users, each pair of users must share a unique key – challenge # 2



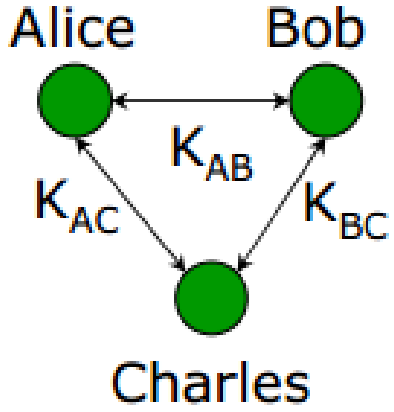
total number of keys required = (N-1)+(N-2)+(N-3)+...+1 = ((N-1)*N)/2

Symmetric Ciphers (cont.)

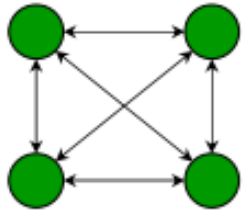
Example: Private-key encryption – number of keys



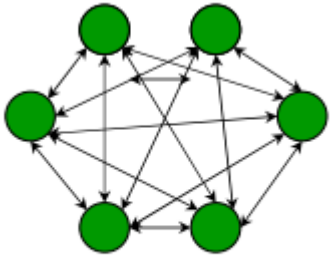
2 users: 1 key



3 users: 3 keys



4 users: 6 keys



6 users: 15 keys

100 users: 4950 keys

1000 users: 499,500 keys

Symmetric Ciphers (cont.)

Example: Symmetric Key Distribution

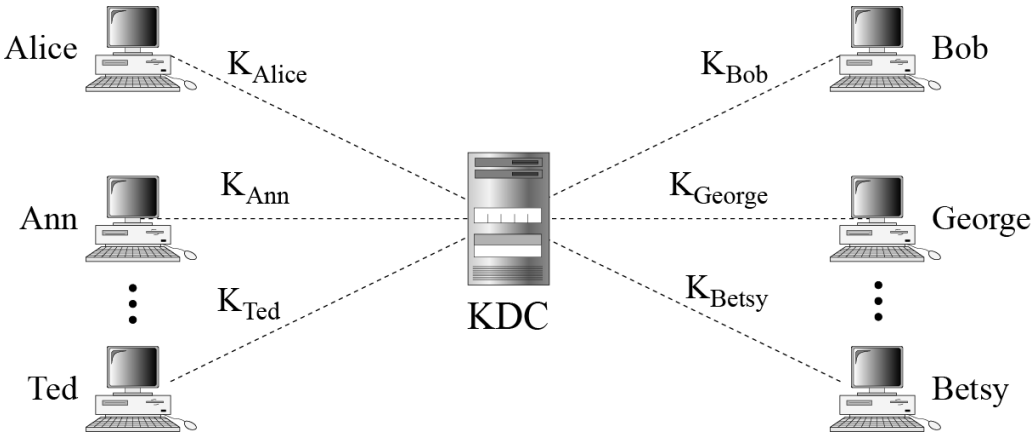
In systems deploying symmetric encryption both the number and distribution of keys is a problem.

Solution: Key Distribution Center (KDC) - trusted 3rd party/server.

Each entity shares a secret key with KDC - N keys in total.

KDC hands out keys to each pair of communicating entities (**M**) on demand, to enable confidential communication between them.

After use, keys are 'recycled'.



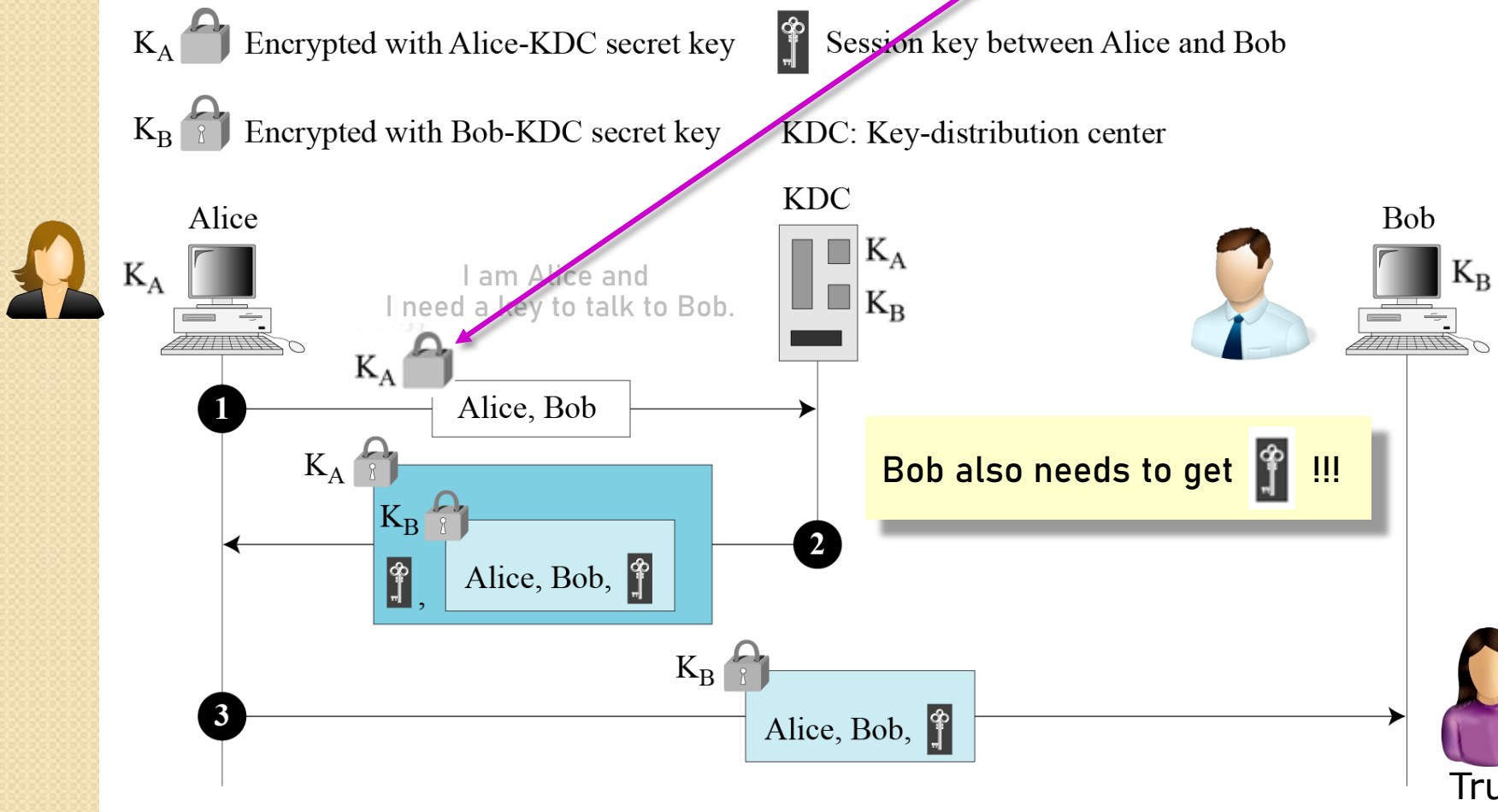
total number of keys in use in the system = = N + M

Symmetric Ciphers (cont.)

Example: Symmetric Key Distribution (cont.)

Possible solution.

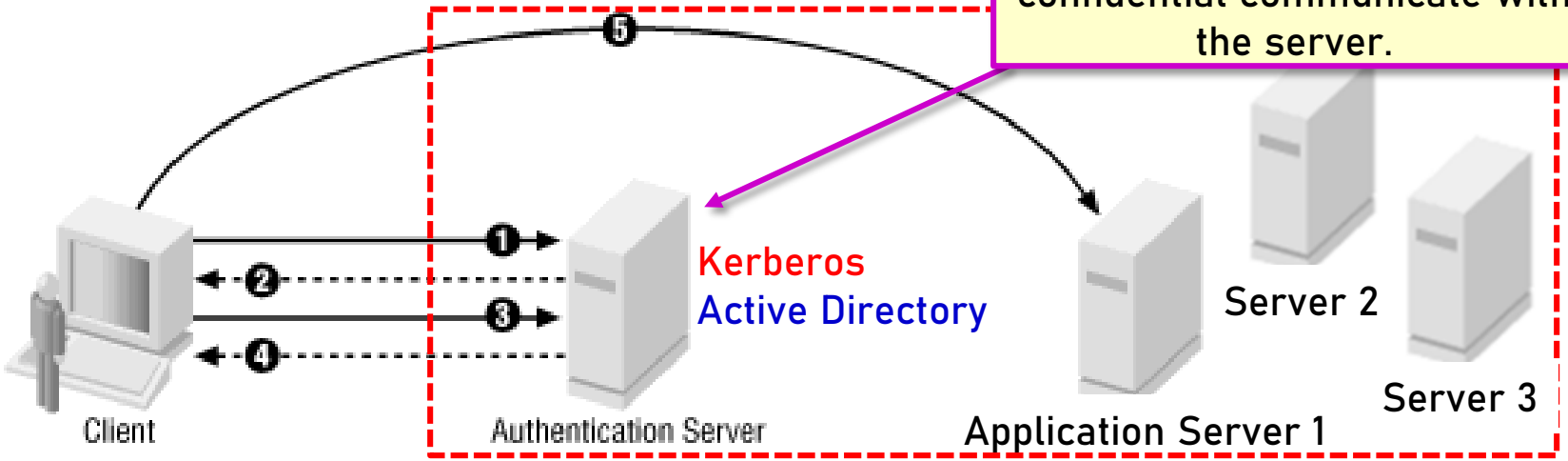
Why is this message encrypted ??



Symmetric Ciphers (cont.)

Example: KDC / Kerberos for SSO

The server does not hold a permanent key for every user. Only after a user properly authenticates himself, he is able to acquire symmetric key for confidential communicate with the server.



1. *Request for ticket granting ticket (TGT)*
2. *TGT returned by authentication service*
3. *Request for application ticket (authenticated with TGT)*
4. *Application ticket returned by ticket-granting service*
5. *Request for service (authenticated with application ticket)*

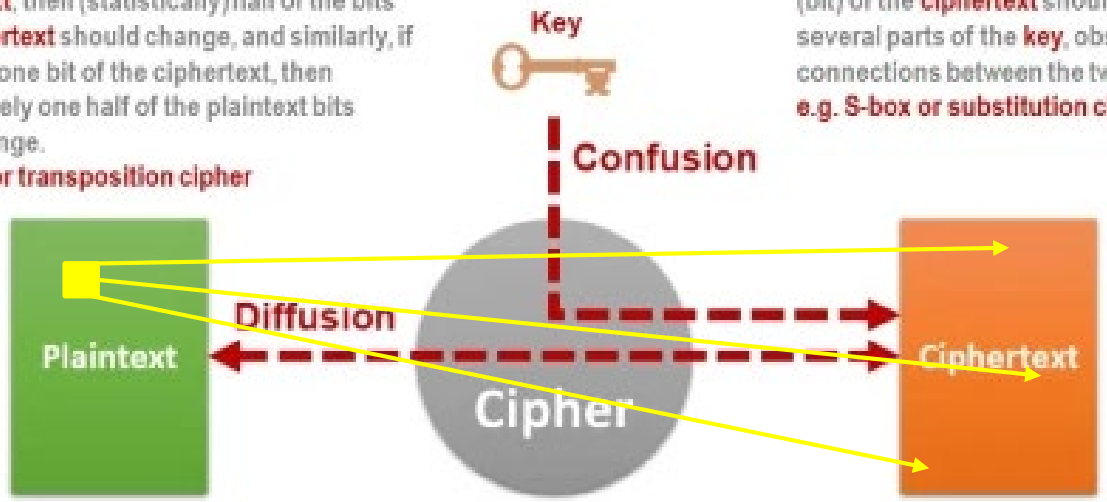
After one single successful authentication with Kerberos server, user can conduct confidential communication with any other server in the system.

Symmetric Ciphers (cont.)

- **Confusion vs. Diffusion** – desired crypto properties ...
  - confusion = making the plaintext-ciphertext substitution (i.e., relationship between the key and the ciphertext) as complex and involved as possible
  - diffusion (permutation) = ensuring that the statistics of the plaintext is dissipated in the statistics of the ciphertext

Diffusion means that if we change a single bit of the **plaintext**, then (statistically) half of the bits in the **ciphertext** should change, and similarly, if we change one bit of the ciphertext, then approximately one half of the plaintext bits should change.
e.g. P-box or transposition cipher

Confusion means that each binary digit (bit) of the **ciphertext** should depend on several parts of the **key**, obscuring the connections between the two.
e.g. S-box or substitution cipher



One block of ciphertext should not depend only on one particular block of plaintext.

If attacker knows just a smaller part of plaintext and ciphertext, he should not be able to 'hack' the entire key.

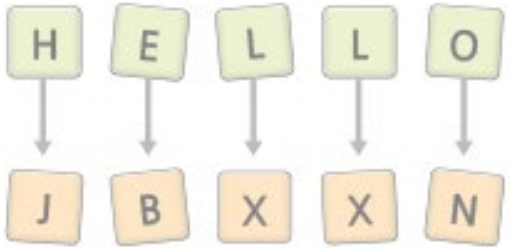
Symmetric Ciphers (cont.)

- categories of Symmetric Encryption:
  - a) **Stream Cipher** – encrypt digits (bytes) of a message one at a time
    - → advantage: speed of transformation – each symbol is encrypted as soon as it is read
    - → disadvantage: low diffusion – all information of a plain-text symbol is contained in a single ciphertext symbol
    - → disadvantage: sensitivity to tampering – an interceptor can splice together pieces of previous messages and transmit a new message that looks authentic
    - → examples: **RC4**, **ChaCha**, **FISH, SEAL, …**

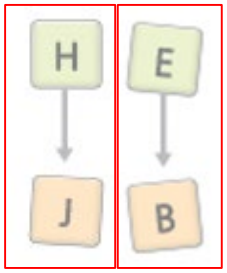
Symmetric Ciphers (cont.)

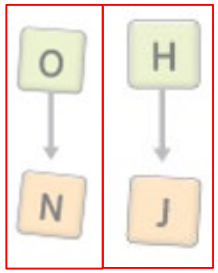
Example: simple message modification attack

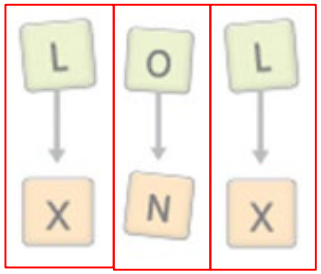
Assume:
- low diffusion algorithm,
- **known plaintext**, or
- **chosen plaintext attack** …

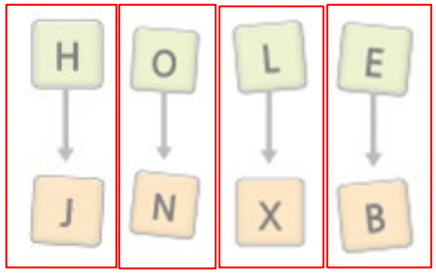


Which words (& respective ciphertext) could Trudy generate/synthesize from this for purposes of injection attack ??



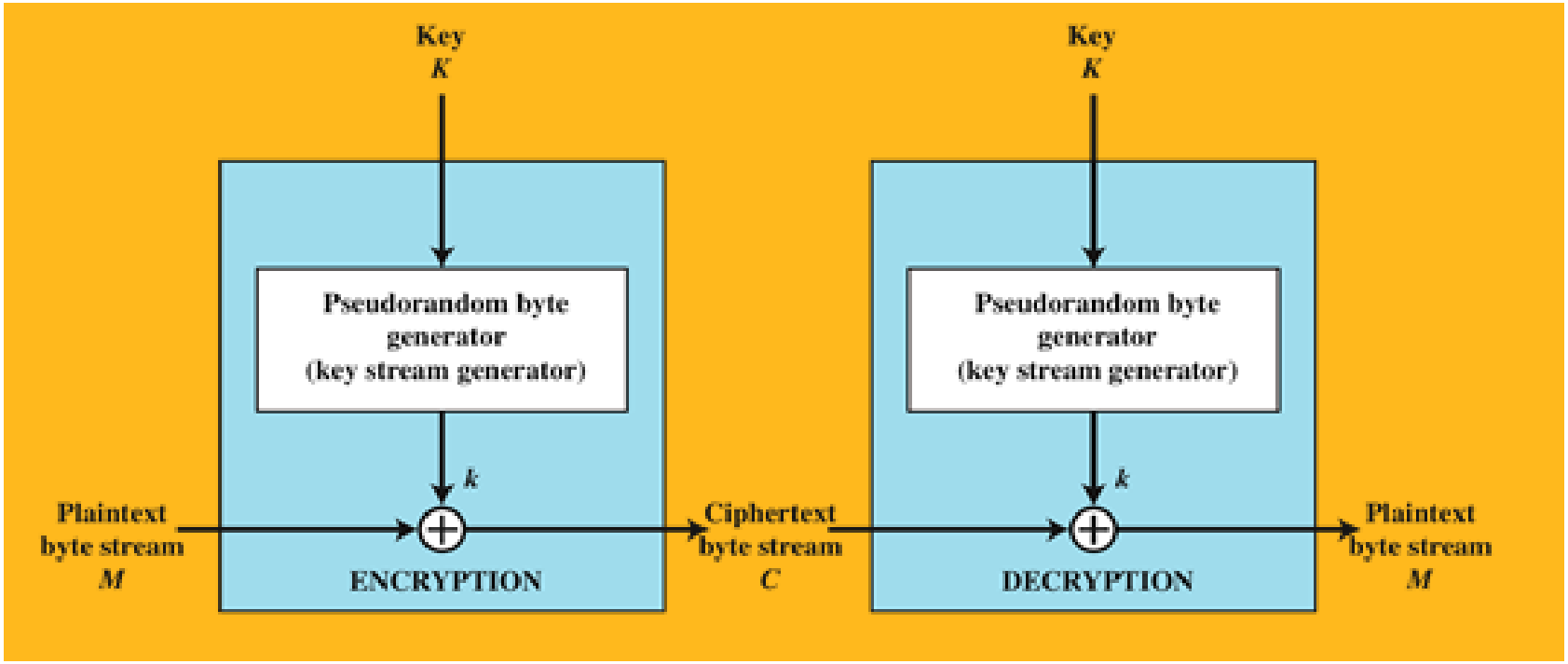






Symmetric Ciphers (cont.)

- categories of symmetric encryption:
  - a) **Stream Cipher** – improvement: pseudo-randomized key



key changes in pseudo-random manner – hard for attacker to predict, yet fully known to communicating parties

Symmetric Ciphers (cont.)

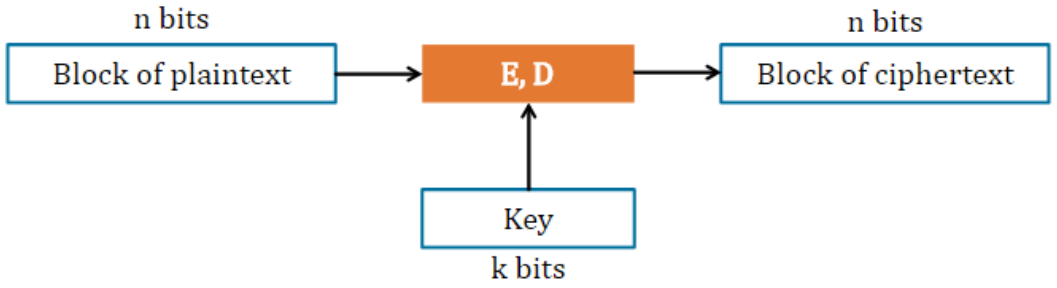
- categories of symmetric encryption (cont.)

b) **Block Cipher** – data is divided into fixed length blocks – all block bits are then acted upon to produce an output

→ advantage: high diffusion – information from one plaintext symbol is diffused into several ciphertext symbols

→ disadvantage: slowness of encryption – an entire block must be accumulated before encryption / decryption can begin => slows down real-time app.

→ examples: **DES**, **3DES**, **AES**



Symmetric Ciphers (cont.)

Example: Symmetric Ciphers in TLS / HTTPS

Cipher			Protocol version						
Type	Algorithm	Nominal strength (bits)	SSL 2.0	SSL 3.0 [n 1][n 2][n 3][n 4]	TLS 1.0 [n 1][n 3]	TLS 1.1 [n 1]	TLS 1.2 [n 1]	TLS 1.3	
Block cipher with mode of operation	AES GCM[57][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	Secure	
	AES CCM[58][n 5]		N/A	N/A	N/A	N/A	Secure	Secure	
	AES CBC[n 6]		N/A	Insecure	Depends on mitigations	Depends on mitigations	Depends on mitigations	N/A	
	Camellia GCM[59][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	N/A	
	Camellia CBC[60][n 6]		N/A	Insecure	Depends on mitigations	Depends on mitigations	Depends on mitigations	N/A	Define
	ARIA GCM[61][n 5]	256, 128	N/A	N/A	N/A	N/A	Secure	N/A	
	ARIA CBC[61][n 6]		N/A	N/A	Depends on mitigations	Depends on mitigations	Depends on mitigations	N/A	
	SEED CBC[62][n 6]	128	N/A	Insecure	Depends on mitigations	Depends on mitigations	Depends on mitigations	N/A	
	3DES EDE CBC[n 6][n 7]	112[n 8]	Insecure	Insecure	Insecure	Insecure	Insecure	N/A	

https://en.wikipedia.org/wiki/Transport_Layer_Security

Example: Real-World Prevalence of Symmetric Ciphers in TLS / HTTPS

TABLE 1: THE MOST POPULAR SELECTED CIPHER SUITES IN THE TOP MILLION SITES

AES is #1 choice for over 97% of sites

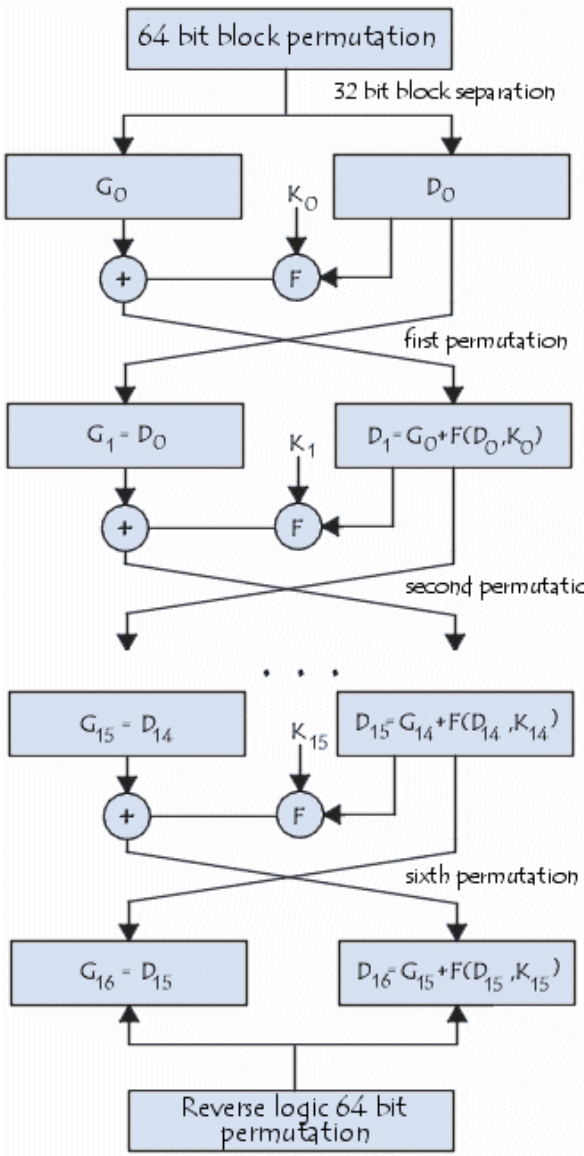
Protocol	Cipher suite chosen by web server	Proportion of top 1M sites
TLS 1.3	TLS_AES_256_GCM_SHA384	56.8%
TLS 1.2	ECDHE-RSA-AES256-GCM-SHA384	18.4%
TLS 1.2	ECDHE-RSA-AES128-GCM-SHA256	12.6%
TLS 1.3	TLS_AES_128_GCM_SHA256	5.4%
TLS 1.2	ECDHE-RSA-AES256-SHA384	1.9%
TLS 1.2	ECDHE-RSA-CHACHA20-POLY1305	1.4%
TLS 1.3	TLS_CHACHA20_POLY1305_SHA256	0.5%
TLS 1.2	ECDHE-ECDSA-AES256-GCM-SHA384	0.4%
TLS 1.2	ECDHE-ECDSA-CHACHA20-POLY1305	0.4%
TLS 1.2	DHE-RSA-AES256-GCM-SHA384	0.4%
TLS 1.2	ECDHE-ECDSA-AES128-GCM-SHA256	0.3%
TLS 1.0	DHE-RSA-AES256-SHA	0.3%

https://www.f5.com/labs/articles/threat-intelligence/the-2021-tls-telemetry-report

Symmetric Ciphers: DES

- **DES** – Data Encryption Standard
  - one of the first widely used symmetric-key block ciphers
  - *initially proposed by IBM (1974), later modified & adopted by US National Bureau of Standards (1977) as an official Federal Information Processing Standard (FIPS)*
  - takes a 64-bit block of plaintext and a 56-bit key to produce a ciphertext block of 64 bits
  - in 1999, Electronic Frontier Foundation managed to break DES in 22 h, 15 min
  - officially retired in 2005
  - 3DES attempted to solve the problem ...

With todays computing powers, DES can be broken within seconds!!!

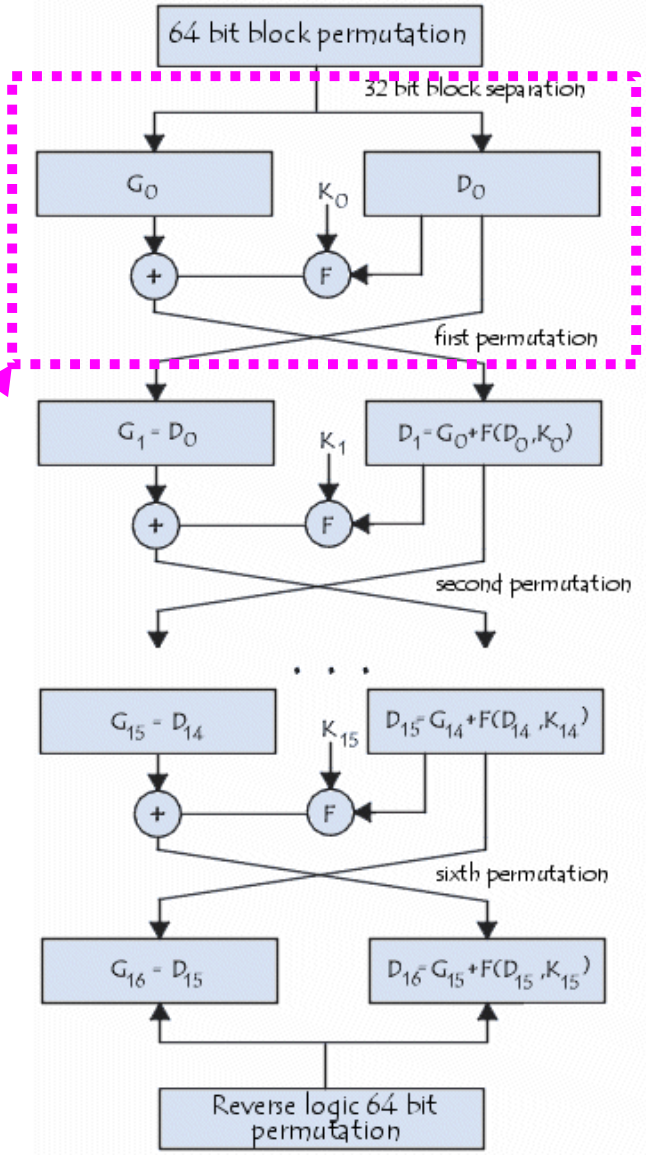


Symmetric Ciphers: DES (cont.)

- DES – Data Encryption Standard
  - **algorithm:**

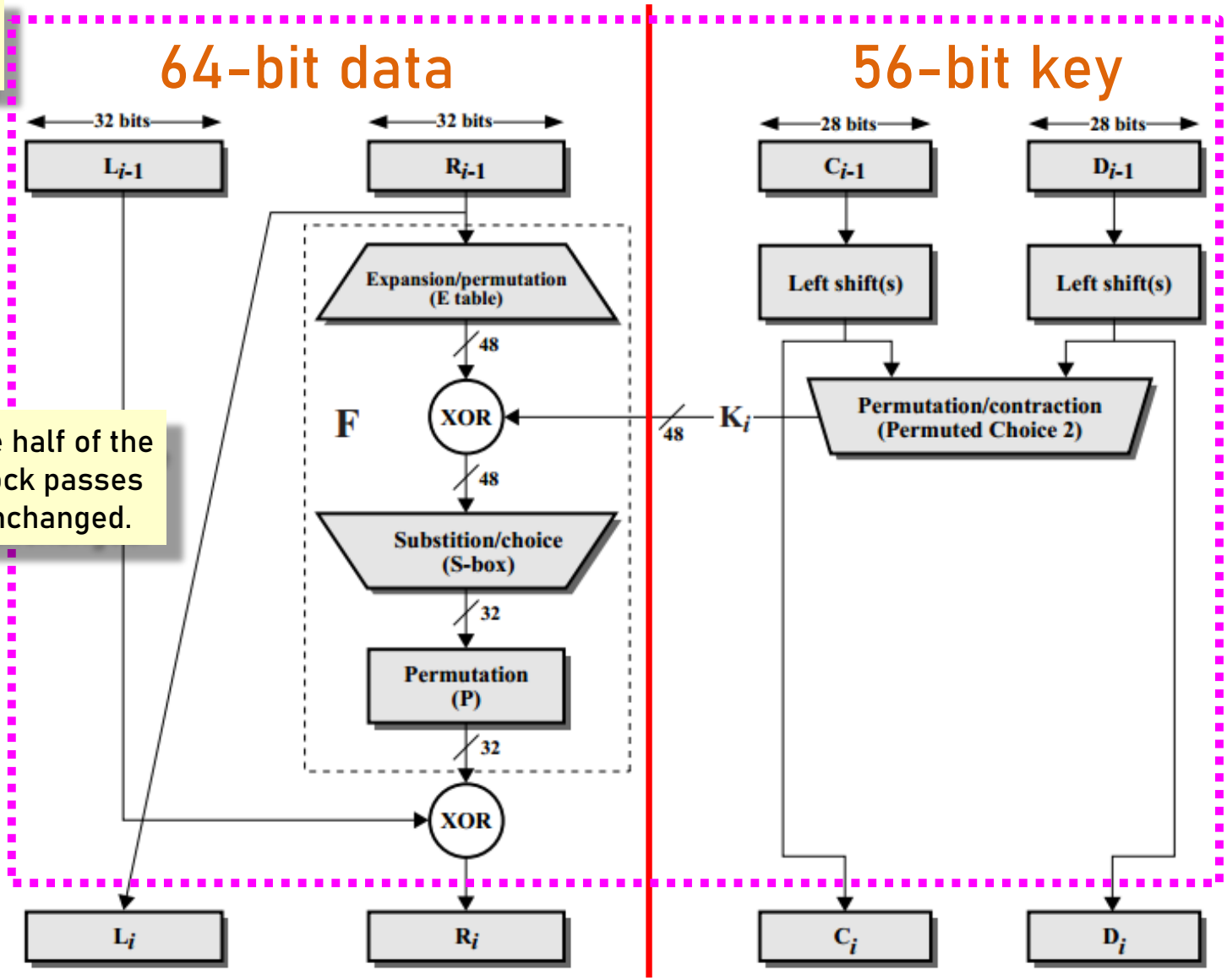
1) plaintext is fractioned into 64-bit locks
2) each block is broken into two parts – left (L) and right (R)
3) permutation and substitution are repeated **16 times/rounds**
4) each round also uses a **48-bit subkey from the original 56-bit key**
5) in the end, two parts are re-joined and undergo inverse initial permutation

In 3DES, there is 3 x 16 rounds of these permutation & substitutions



Symmetric Ciphers: DES (cont.)

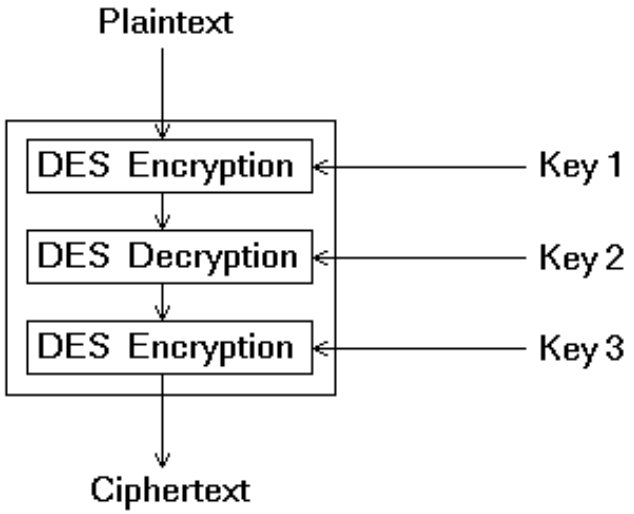
ONE SINGLE ROUND OF DES !!!



One half of the block passes unchanged.

Symmetric Ciphers: 3DES

- **Triple DES = TDES = 3DES**
  - symmetric-key block cipher which applies DES 3 times to each data block = Encrypt + Decrypt + Encrypt



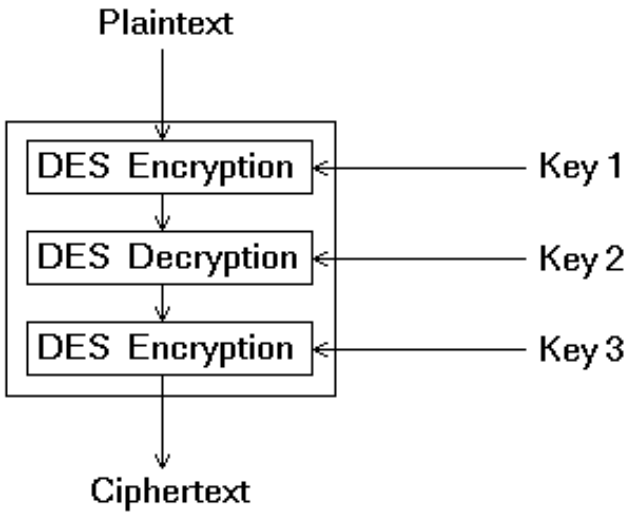
Ciphertext = $E_{K3}(D_{K2}(E_{K1}(\text{Plaintext})))$

  - proposed in 1978, accepted as FIPS in 1999
  - a simple method of strengthening (increasing key size of) DES, without the need to design a completely new algorithm
  - current use – electronic payment industry (until 2023!)

Symmetric Ciphers: 3DES (cont.)

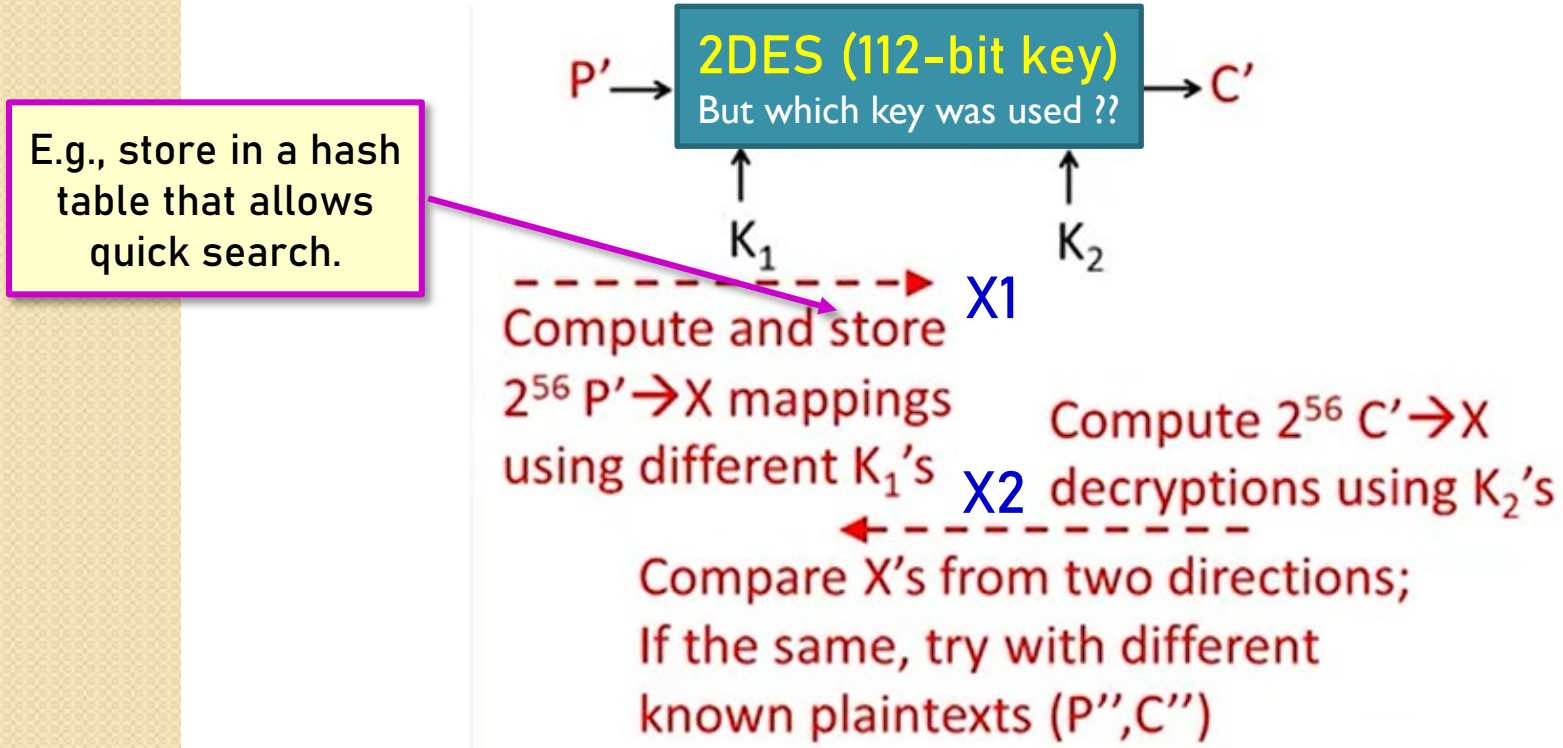
- **Triple DES Keying Options**
  - Option 1: all three keys are independent
    - * total key size = 168 bits
    - * effective security = 112 bits
    - * strongest
  - Option 2: K1 and K2 are independent, K3=K1
    - * total key size = 112 bits
    - * effective security = 80 bits
    - * retired in 2015
  - Option 3: all three keys the same K1=K2=K3
    - * total key size = 56 bits
    - * weak – just a 'very slow' version of regular DES
    - * not approved

Due to Meet-in-the-Middle attack !!!



Meet-in-the-Middle Attack on 2DES

- theoretical brute-force complexity: 2x56=112-bit key space
- applies to any block-cipher that is sequentially processed (i.e., attempts to increase 'strength' by adding multiple components/stages)
  - * instead of focusing only on input/plaintext & output/ciphertext of entire chain/system, transitional value(s) between components are utilized
- attack works only if a known plaintext-ciphertext is given !!



Symmetric Ciphers: 3DES (cont.)

Triple DES – Pros and Cons

- 3DES, key option 1, still in use, but will be deprecated in 2023
  - many devices in the financial industry (e.g., POS terminals) as well as networking equipment (e.g., firewalls) use 3DES and are challenging to upgrade

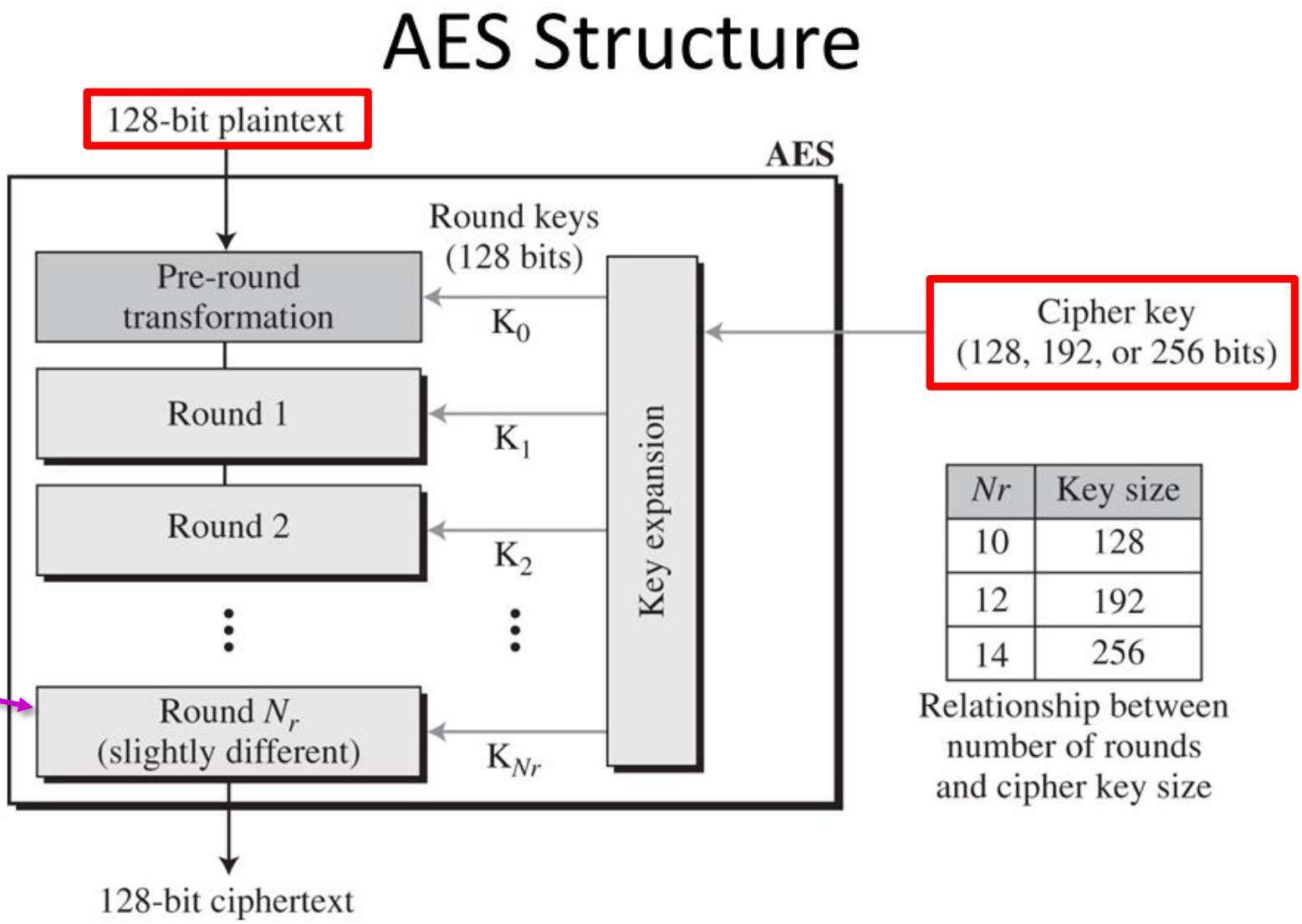
- DES was designed for efficient hardware implementation - software implementation is very slow, 3DES even slower ☹
- DES and 3DES use 64-bit block size – to improve efficiency and security larger block sizes would be preferable ☹

overcome with AES

Symmetric Ciphers: AES

- **AES** – Advanced Encryption Standard
  - NIST issued call for a 3DES replacement in 1997 with requirements:
    - * symmetric block cipher
    - * block size 128
    - * key lengths 128, 192 or 256
  - initially 15, then 5 competing standards were evaluated
  - Rijndael cipher was selected as the most suitable for AES
  - AES became a US FIPS in November 2001
  - AES is intended to replace 3DES, but this process is taking longer than expected ...

Symmetric Ciphers: AES (cont.)



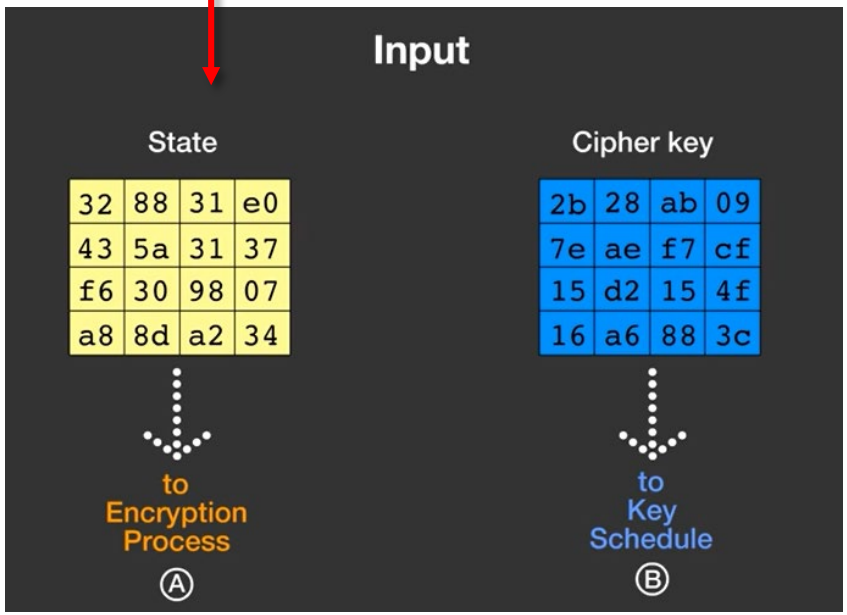
Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Except for the last round, other rounds are identical.

Symmetric Ciphers: AES (cont.)

128 bits = 4x4 bytes



The 4 types of transformations:

1-SubBytes

2-ShiftRows

3-MixColumns

4-AddRoundKey

https://www.youtube.com/watch?v=gP4PqVGudtg

Symmetric Ciphers: AES (cont.)

AES Facts

- Like DES, AES is an iterated block cipher in which a block of plaintext is subject to multiple rounds of processing, with each round applying the same overall function.

- Unlike DES, AES applies transformation operation to the entire incoming block in each iteration, while in DES one-half of incoming block passes unchanged.

- Unlike DES which is bit-oriented, AES is byte-oriented $\Rightarrow$ allows convenient and fast software implementation.

- Unlike DES, where 1/64 bits of a plaintext affected roughly 31/64 bits of the ciphertext, in AES (due to shift-row and mix-column steps) each bit of the plaintext affects every bit of the ciphertext.

Symmetric Ciphers: AES (cont.)

No. of Years to crack AES with 128-bit Key = $(3.4 \times 10^{38}) / [(10.51 \times 10^{12}) \times 31536000]$

$= (0.323 \times 10^{26})/31536000$

$= 1.02 \times 10^{18}$

= 1 billion billion years

Key size	Time to Crack
56-bit	399 seconds
128-bit	1.02×10^{18} years
192-bit	1.872×10^{37} years
256-bit	3.31×10^{56} years

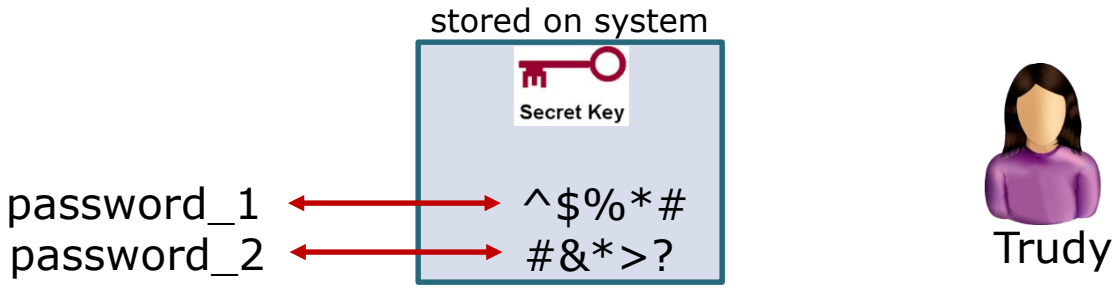
Figure 4: Time to crack Cryptographic Key versus Key size

As shown above, even with a supercomputer, it would take 1 billion billion years to crack the 128-bit AES key using brute force attack. This is more than the age of the universe (13.75 billion years). If one were to assume that a computing system existed that could recover a DES key in a second, it would still take that same machine approximately 149 trillion years to crack a 128-bit AES key.

http://www.eetimes.com/document.asp?doc_id=1279619 - July 2012

Practical Uses of Symm. Encrypt.

- **How to protect passwords on/in a system ...**



Is use of symmetric encryption
with a single master encryption key
a good way to
protect passwords in a system ??

Practical Uses of Symm. Encrypt.

Example: Target and 3DES

On Dec. 23, 2013, Target confirmed malware was to blame for an infection of its point-of-sale system that likely exposed details associated with 40 million debit and credit cards (50GB of encrypted data) between Nov. 27 and Dec. 15.

In its statement, Target notes that:
"The most important thing for our guests to know is that their debit card accounts have not been compromised due to the encrypted PIN numbers being taken."
"... PINs are encrypted at the keypad with what is known as Triple DES" - a standard the retailer refers to as being highly secure and used broadly throughout the U.S.

"Most people object to 3DES because it's an ancient algorithm that was designed as a patch for (now broken) DES until AES was finalized," ... "Now we've had AES for more than a decade, it's questionable why we'd be using 3DES."

https://threatpost.com/targets-use-of-3des-encryption-invites-scrutiny-worry/103389

Practical Uses of Symm. Encrypt.

Example: Encrypting PIN Pad

An **Encrypting PIN Pad** is an apparatus for encrypting an identifier such as a PIN as soon as it is entered on a keypad. These are used in ATM and POS terminals to ensure that the unencrypted PIN is not stored or transmitted anywhere in the rest of the system and thus cannot be revealed accidentally or through manipulations of the system.

'Master Key' for terminal is either manually set or remotely loaded.

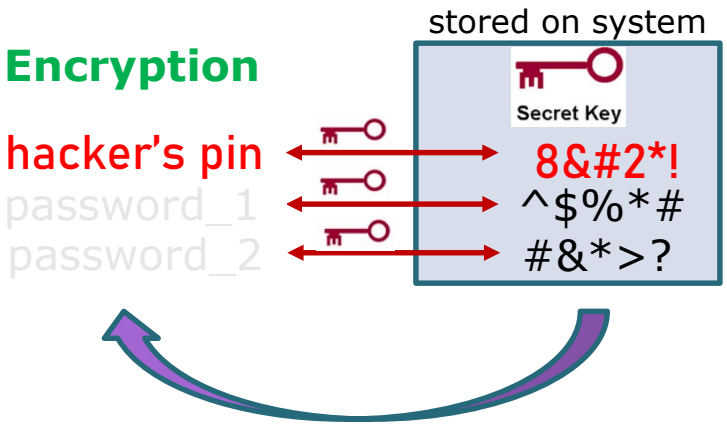
https://en.wikipedia.org/wiki/Encrypting_PIN_Pad

Practical Uses of Symm. Encrypt.

Example: Target and 3DES

Should passwords be encrypted?

Encryption



assume ciphertext only attack

3DES decryption is time consuming as it requires the search through 168-bit key space!

Plus, passwords are hard to validate (likely not plain English words).

But, what if 'chosen plaintext' attack is conducted ??

If hacker knows one pin (e.g., his own) and its respective ciphertext, he can conduct (faster) Meet-in-the-Middle attack, and once he finds the key, he can crack all other pins from the same POS device!

Symmetric Ciphers: 3DES (cont.)

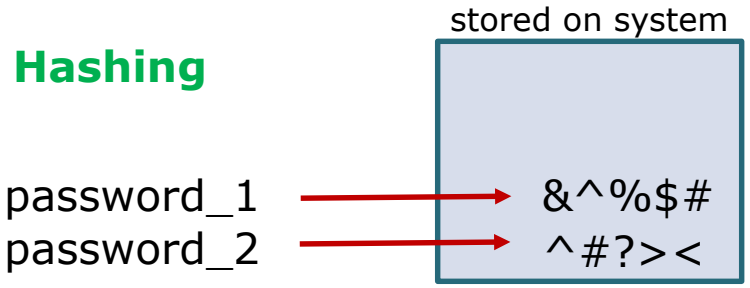
Example: Target and 3DES

Should passwords be encrypted?

	Hashing	Symmetric Encryption
	One-way function	Reversible Operation
Invertible Operation?	No,	Yes,
	For modern hashing algorithms it is not easy to reverse the hash value to obtain the original input value	Symmetric encryption is designed to allow anyone with access to the encryption key to decrypt and obtain the original input value

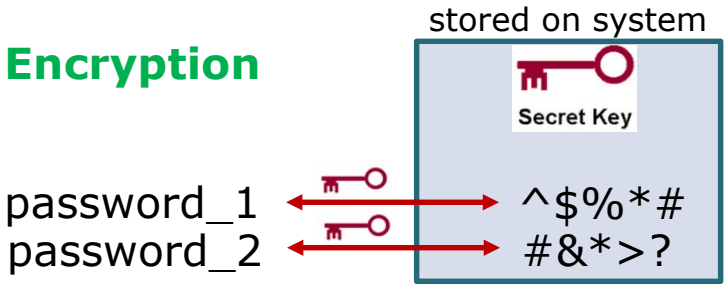
http://www.darkreading.com/safely-storing-user-passwords-hashing-vs-encrypting/a/d-id/1269374

Hashing



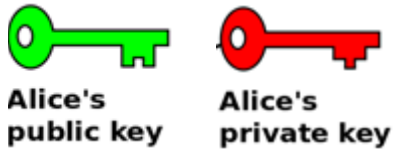
cracking one password does not assist in cracking other passwords – passwords have to be cracked 'one by one'

Encryption

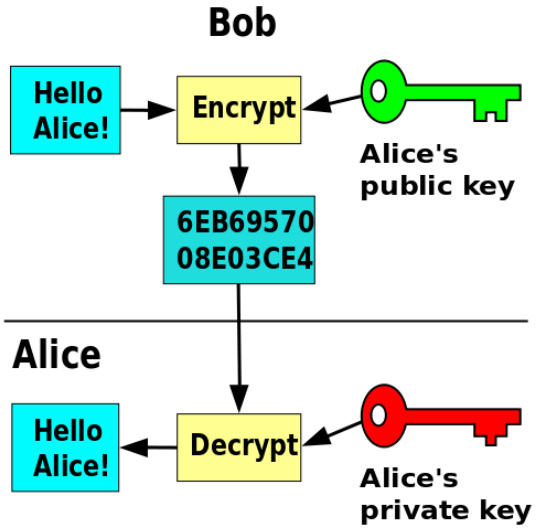


obtaining the key or cracking one password expedites cracking of all other passwords

Asymmetric Ciphers

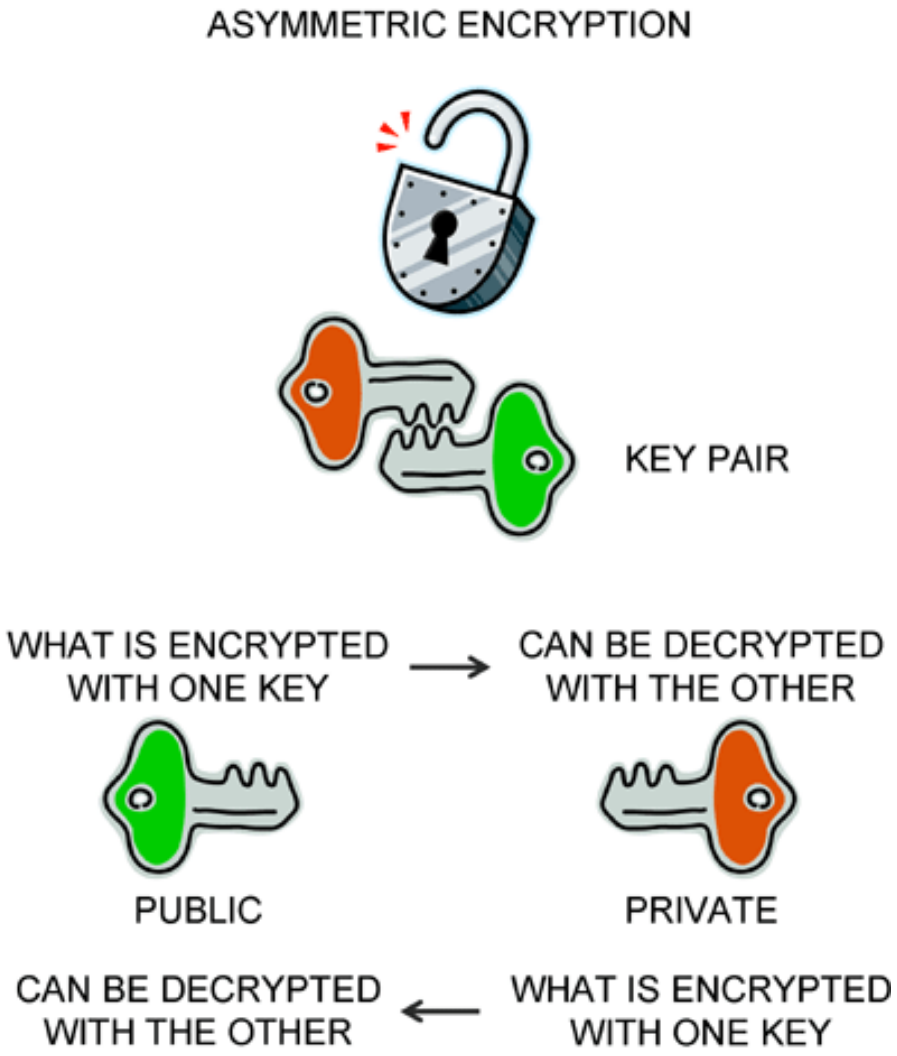


- **Asymmetric Encryption** – aka Public-Key Encryption – involves the use of two separate but related keys: public key and private key



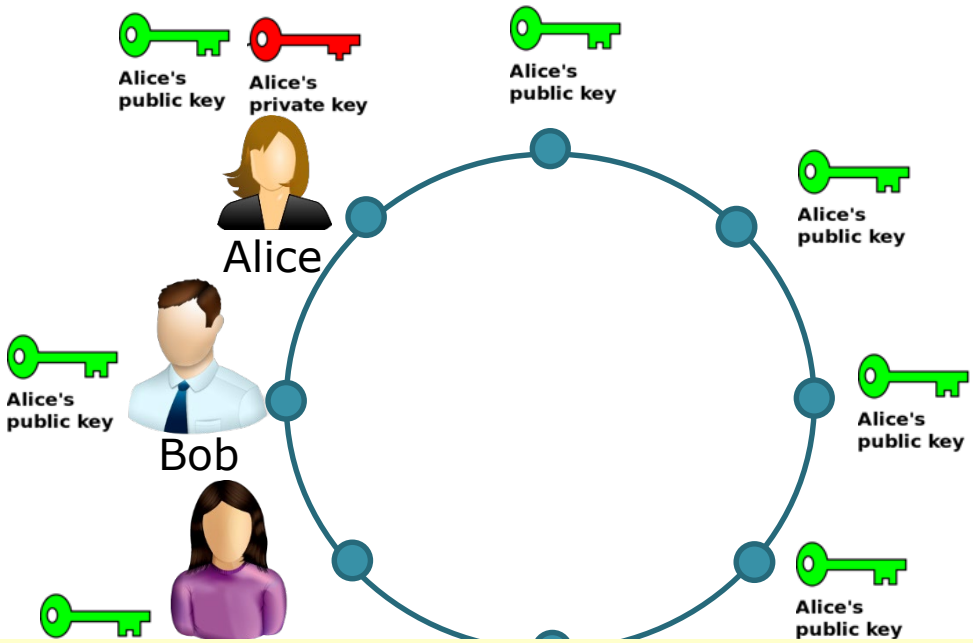
  - public key is made public for others to use, private key is known only to its owner
  - either key can encrypt a message – the other key must be used for decryption
  - first truly revolutionary advance in encryption, with profound consequences in the areas of
    * confidentiality
    * authentication
    * key distribution

Asymmetric Ciphers (cont.)



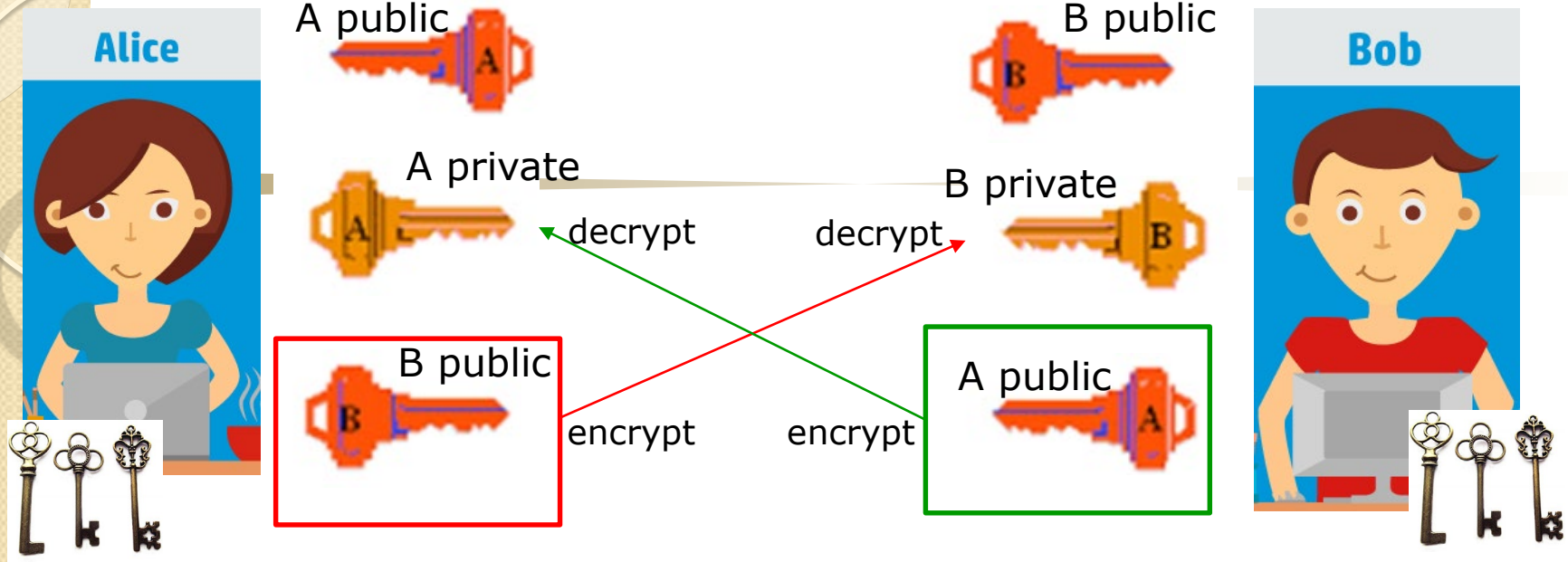
Asymmetric Ciphers

Example: Keys in asymmetric cipher system ...



Public key is sent only to other people/entities with whom Alice wants to confidentially communicate !!!

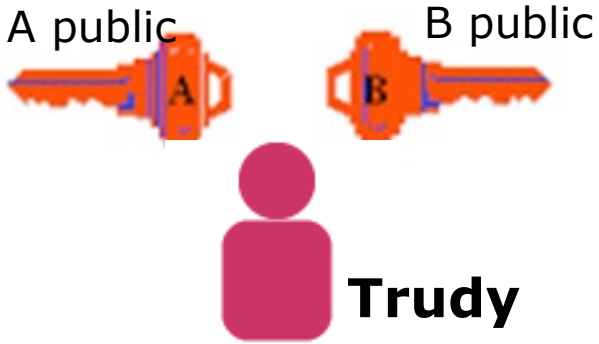
The overall number of different keys generated (in the 'existence'):
$O(2*N) = O(N) <<< O(N^2)$



What key should Alice use to

a) Send a confidential message to Bob???

b) Receive a confidential message from Bob???



Asymmetric Ciphers (cont.)

- **Asymmetric Encryption: Mode 1.a)**

Protection of Confidentiality: Alice receives message from Bob

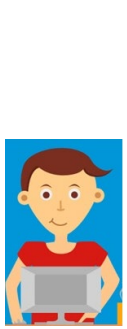
(1) Each user generates a pair of keys.

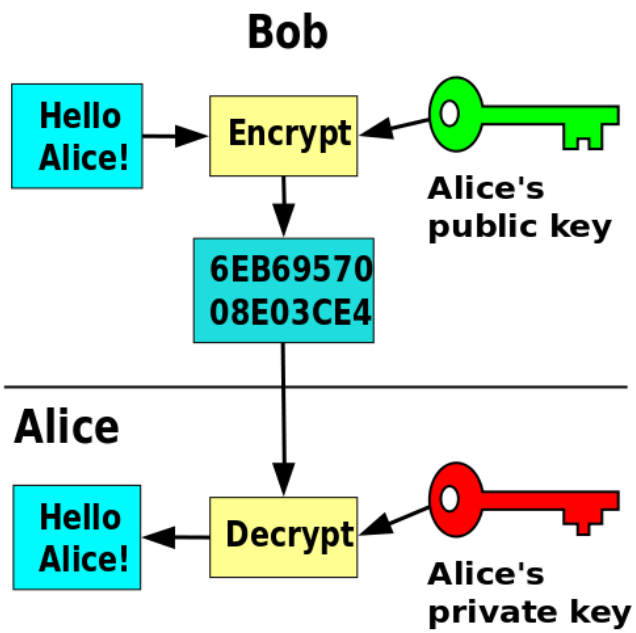
(2) Each user places one of the keys in a public register - this becomes the public key; the other is private key.

(3) If Bob wishes to send a private message to Alice, he uses Alice's public key.

(4) To decrypt Bob's message, Alice uses her private key.

No other recipient can decrypt Bob's message as only Alice knows her key.

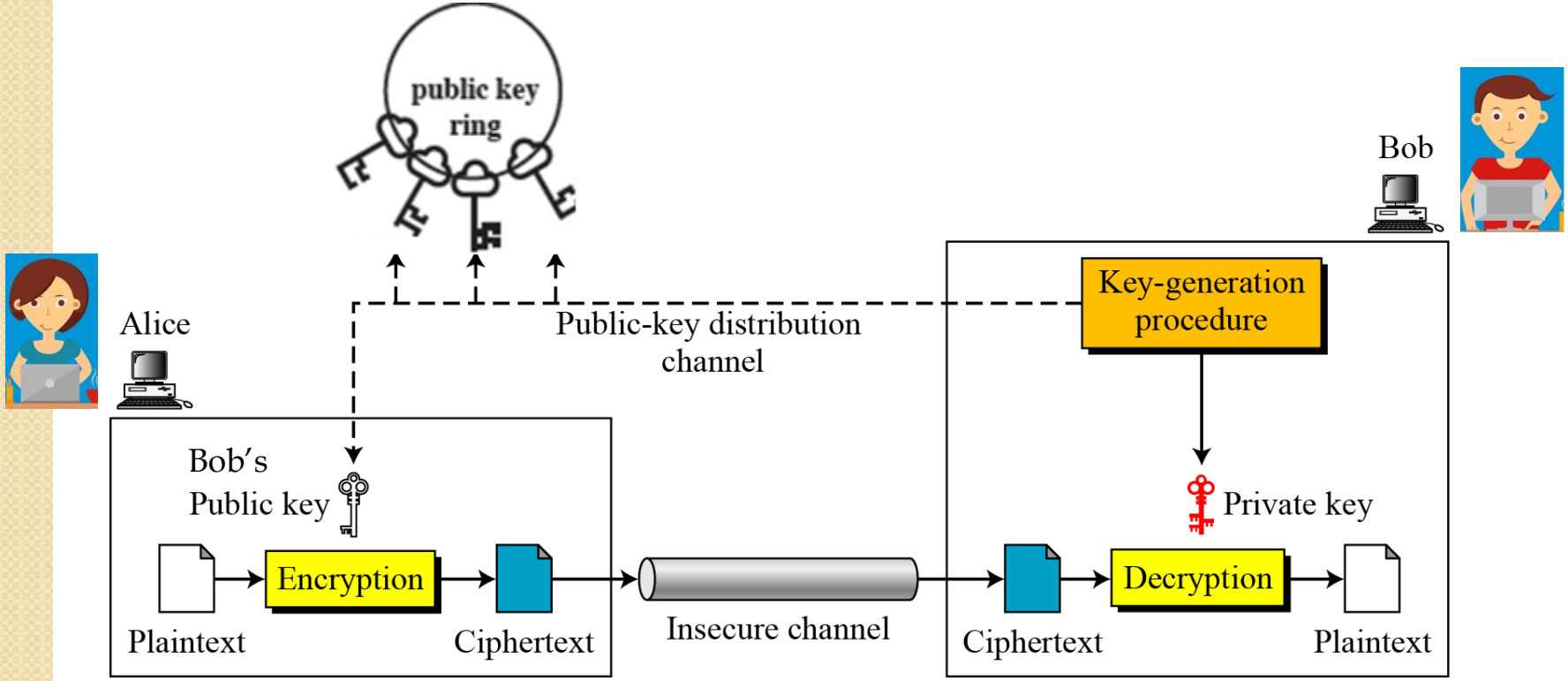




Asymmetric Ciphers (cont.)

Example: Asymmetric Encryption: Mode 1.b)

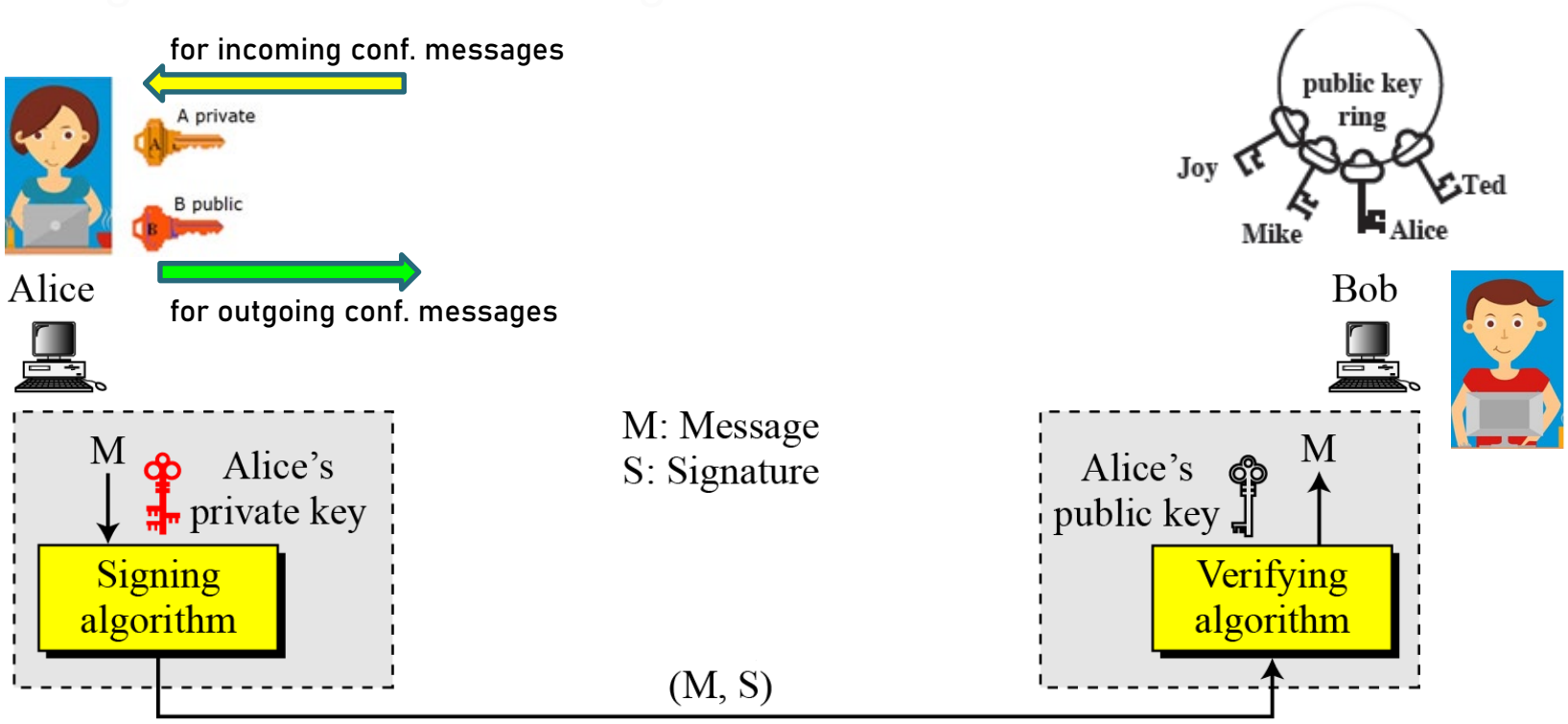
Protection of Confidentiality: Alice sends message to Bob



Asymmetric Ciphers (cont.)

Example: Asymmetric Encryption: Mode 2

Is there any merit in Alice sending a message to Bob signed with her private key ???



Asymmetric Ciphers (cont.)

- **Symmetric vs. Asymmetric Encryption** – common misconceptions

  (1) **public-key encryption is a general-purpose technique that has made symmetric encryption obsolete**

  * public-key encryption is versatile but very slow – **symmetric encryption is still needed for encryption of large messages**!

  * public-key encryption is used for authentication, digital signatures, and exchanges of secret keys!

  (2) **exchange of asymmetric/public keys is much simpler than exchange of symmetric/secret keys**

  * both schemes require a well established system and protocols

Asymmetric Ciphers (cont.)

- **Symmetric vs. Asymmetric Encryption** (cont.)

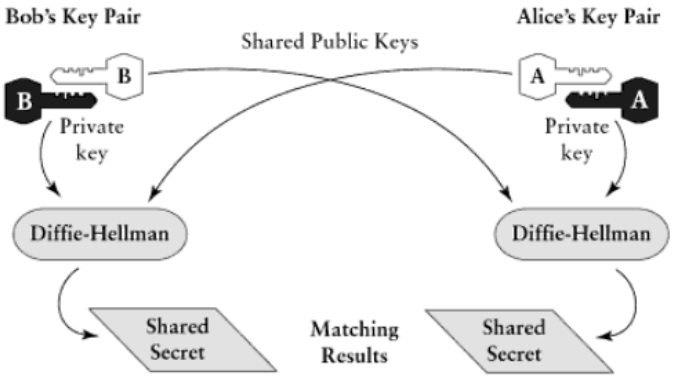
TABLE 8.1 Comparison of secret-key and public-key crypto

Type	Secret Key	Public Key
Symmetry	Symmetric	Asymmetric
Number of keys	There is one crypto key; the sender and recipient both use it.	There are two crypto keys; the sender uses one and the recipient uses the other.
Key secrecy	The *secret key* is always kept secret.	The *private key* is kept secret. The *public key* is published and shared.
Calculating efficiency	Requires a great deal of calculation using relatively small numbers.	Requires a great deal of calculation using very, very large numbers.
Typical key sizes	128–256 bits	1024–4096 bits
Unique identities	Every member of the cryptonet uses the same key; it is impossible to distinguish one member from another.	Each user may have a unique private key; only the corresponding public key works with a given private key.

Elementary Information Security, R. E. Smith, pp. 347

Asymmetric Ciphers: D-H

- **Diffie-Hellman** – first published public-key encryption algorithm (1976)
  - currently used in **TLS** (Transport Layer Security), **SSH** **IPSec** protocol
  - purpose: enable two users to securely reach agreement (i.e., generate) a secret key for subsequent symmetric encryption without the involvement of a Key Dist. Cent. (KDC)
  - property: private key A and public key B generate the same result as private key B and public key A



Asymmetric Ciphers: D-H (cont.)

- **Diffie-Hellman** – the basics of the math ...

(1) Before establishing a symmetric key, two parties choose/obtain two integer numbers:
p – large prime number with 1024 bits (300 decimal digits)
g – base or generator (primitive root of mod p) – often 2, 3, 7

(2) Alice chooses a large random number **x** $(1 \le x \le p-1)$ and calculates $R_x = g^x \bmod p$.
x: Alice's private key
R_x: Alice's public key

(3) Bob chooses another large random number **y** $(1 \le y \le p-1)$ and calculates $R_y = g^y \bmod p$.
y: Bob's private key
R_y: Bob's public key

(4) Alice sends Bob R_x, Bob sends Alice R_y.

(5) Alice calculates $K = (R_y)^x \bmod p$.

(6) Bob calculates $K = (R_x)^y \bmod p$.

$$K = (g^y \bmod p)^x \bmod p = (g^x \bmod p)^y \bmod p = g^{xy} \bmod p$$

NOT REQUIRED !!!

Diffie-Hellman key exchange algorithm

The Diffie-Hellman key exchange uses *primitive roots* modulo a prime.

Primitive roots

Let p be a prime number. A positive number a is a *primitive root modulo* p if the positive powers $a^1, a^2, \ldots, a^{p-1}$ generate all nonzero congruence classes modulo p. For example, let $p = 7$. Then $a = 2$ is not a primitive root, since

$$2^1 = 2, \quad 2^2 = 4, \quad 2^3 \equiv 1, \quad 2^4 \equiv 2, \quad 2^5 \equiv 4, \quad 2^6 \equiv 1 \pmod 7.$$

The powers of 2 never hit the classes $3, 5$, or 6 modulo 7. However, 3 is a primitive root since

$$3^1 = 3, \quad 3^2 \equiv 2, \quad 3^3 \equiv 6, \quad 3^4 \equiv 4, \quad 3^5 \equiv 5, \quad 3^6 \equiv 1 \pmod 7.$$

That is, each congruence class $\pmod 7$ appears as a power of 3.

http://www.acme.byu.edu/wp-content/uploads/2014/09/Diffie-Hellman.pdf

Asymmetric Ciphers: D-H (cont.)

Example: Who knows what ....

Alice		Bob		Eve	
Known	Unknown	Known	Unknown	Known	Unknown
$p = 23$		$p = 23$		$p = 23$	
$g = 5$		$g = 5$		$g = 5$	
$a = 6$	b	$b = 15$	a		a, b
$A = 5^a \bmod 23$		$B = 5^b \bmod 23$			
$A = 5^6 \bmod 23 = 8$		$B = 5^{15} \bmod 23 = 19$			
$B = 19$		$A = 8$		$A = 8$, $B = 19$	
$s = B^a \bmod 23$		$s = A^b \bmod 23$			
$s = 19^6 \bmod 23 = 2$		$s = 8^{15} \bmod 23 = 2$			s

A's private key

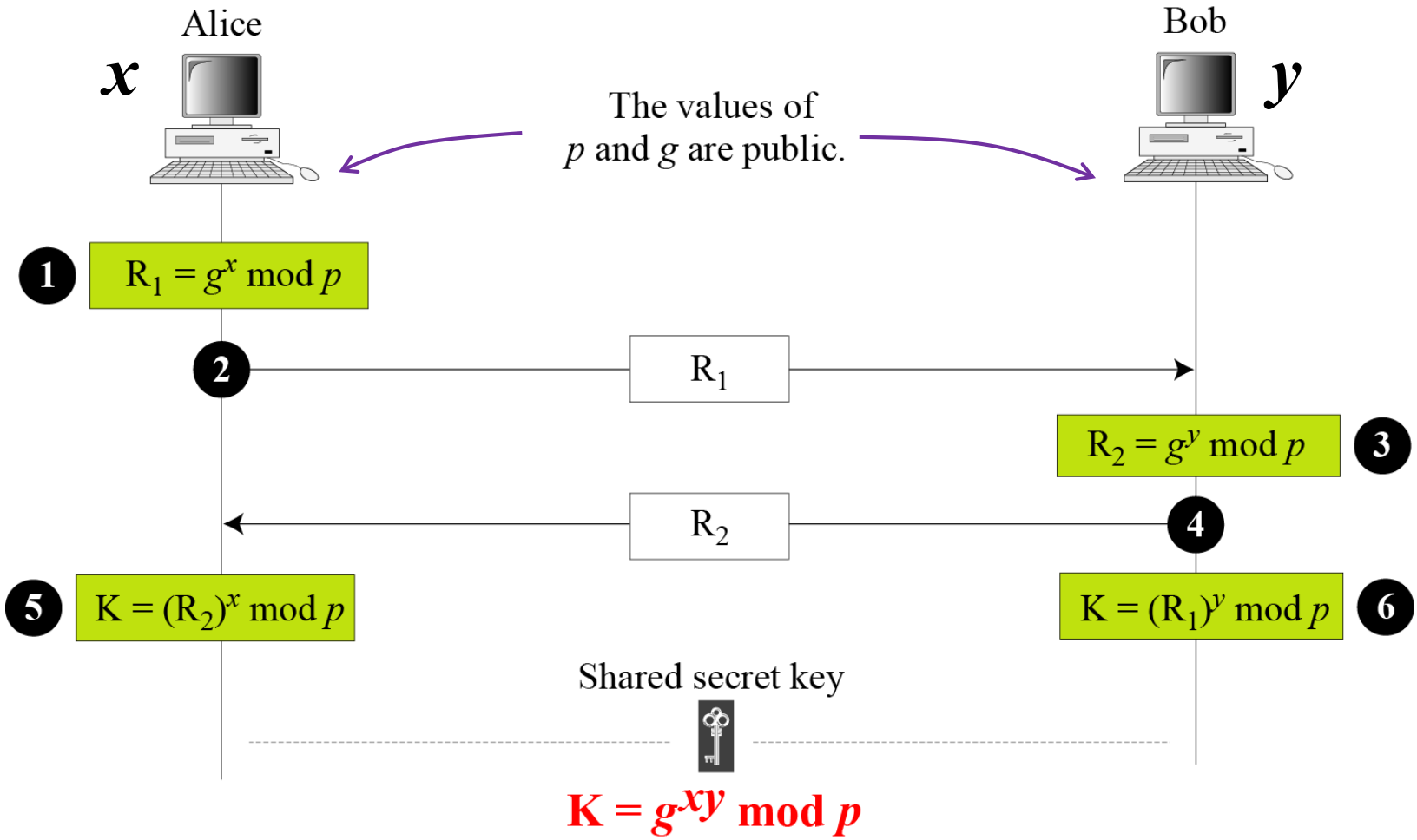
A's public key

B's public key

jointly generated secret symmetric key

Asymmetric Ciphers: D-H (cont.)

Example: Diffie-Hellman exchange



Asymmetric Ciphers: D-H (cont.)

Example: Diffie-Hellman key calculation

Assume that $p = 23$ and $g = 7$.

1. Alice picks $x = 3$ and calculates $R_1 = 7^3 \bmod 23 = 21$.
2. Bob picks $y = 6$ and calculates $R_2 = 7^6 \bmod 23 = 4$.
3. Alice sends the number 21 to Bob.
4. Bob sends the number 4 to Alice.
5. Alice calculates K = $4^3 \bmod 23 = 64 \bmod 23 = 18$.
6. Bob calculates K = $21^6 \bmod 23 = 85766121 \bmod 23 =$ $= 18$.
7. The value of K is the same for both Alice and Bob. $g^{xy} \bmod p = 7^{18} \bmod 23 = 18$.

Assume that $p = 7$ and $g = 2$.

Alice chooses $a = ??$

Bob

$A = g^a \bmod p = 1$

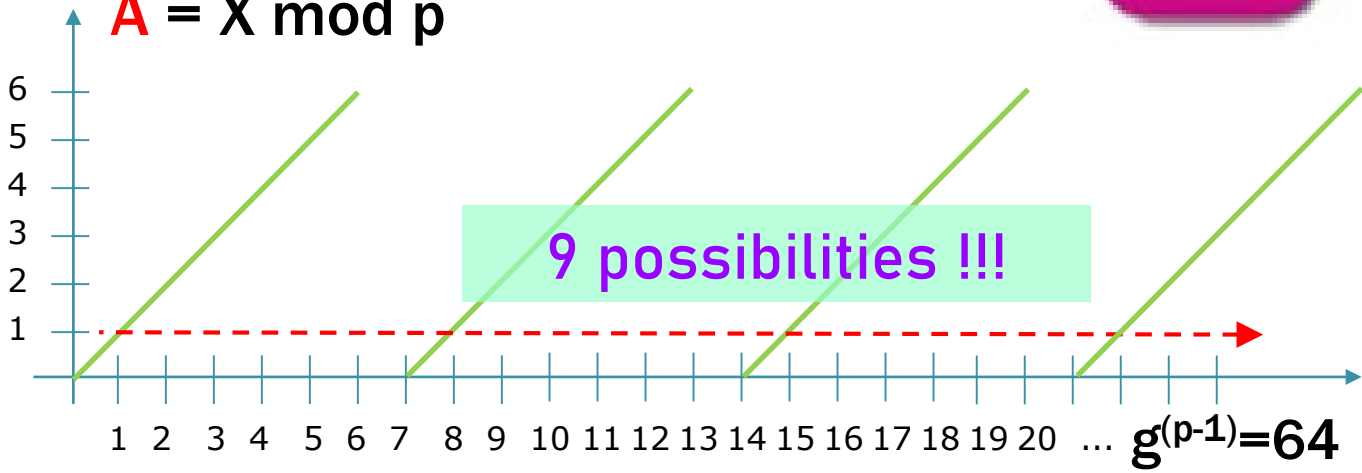
Is it really so hard to determine what a is ???

Assume that $p = 7$ and $g = 2$. Alice chooses $a = 3$.

Assume that $p = 7$ and $g = 2$.

A = 1

A = X mod p

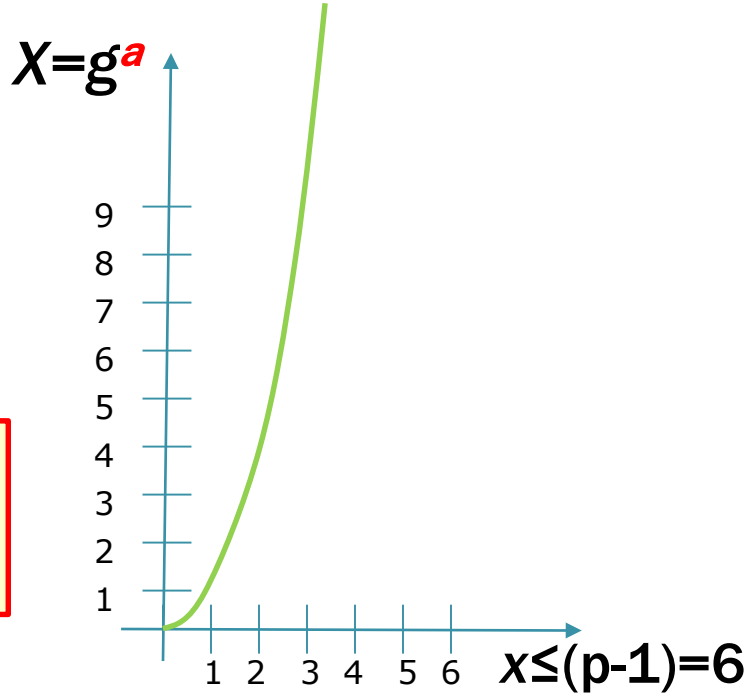


9 possibilities !!!

Is that number of the form: 2^x ??

If it is, check if x is (possibly) the key??

Breaking DH algorithm is computationally hard/infeasible for large p.



Asymmetric Ciphers: D-H (cont.)

Example: Diffie-Hellman – more realistic example:
p is 512 bits long

p	76462429856349357218249376595503050747633809672694974892357377286092523566666075542363742330966118003333810619473013095041473870099917804365478785807987581
g	2
x	557
y	273

R_1	84492028420566550521617294749103509414343369852001266086286363106767361995928082858670080213185929094514021750031997331294583608382194306596602015795535 4
R_2	43526283870920037947074711489558162763638911626211555797512337921856631001143571820839004018187648684175383116534269163026342110672150858962552012885941 43
K	15563800066452229059622582752327076527321804694442367852032040014640650088793665120425742677660832791101715303867456125221315161097658420012040864336177 40

Asymmetric Ciphers: D-H (cont.)

Example: How big is 64-bit int ??

9,223,372,036,854,775,807

The number 9,223,372,036,854,775,807, equivalent to the hexadecimal value $7FFF,FFFF,FFFF,FFFF_{16}$, is the maximum value for a 64-bit signed integer in computing. It is therefore the maximum value for a variable declared as a long integer (`long` , `long long int` , or `bigint`) in many programming languages running on modern computers.[1][2][3]

How is NSA breaking so much crypto?

OCTOBER 14, 2015 BY ALEX HALDERMAN AND NADIA HENINGER

There have been rumors for years that the NSA can decrypt a significant fraction of encrypted Internet traffic. In 2012, James Bamford published **an article** quoting anonymous former NSA officials stating that the agency had achieved a "computing breakthrough" that gave them "the ability to crack current public encryption." The Snowden documents also hint at some extraordinary capabilities: they show that NSA has built extensive infrastructure to intercept and decrypt VPN traffic and suggest that the agency can decrypt at least some HTTPS and SSH connections on demand.

However, the documents do not explain *how* these breakthroughs work, and speculation about possible backdoors or broken algorithms has been rampant in the technical community. Yesterday at ACM CCS, one of the leading security research venues, we and twelve coauthors presented **a paper that we think solves this technical mystery**.

The key is, somewhat ironically, Diffie-Hellman key exchange, an algorithm that we and many others have advocated as a defense against mass surveillance. Diffie-Hellman is a cornerstone of modern cryptography used for VPNs, HTTPS websites, email, and many other protocols. Our paper shows that, through a confluence of number theory and bad implementation choices, many real-world users of Diffie-Hellman are likely vulnerable to state-level attackers.

For the nerds in the audience, here's what's wrong: If a client and server are speaking Diffie-Hellman, they first need to agree on a large prime number with a particular form. There seemed to be no reason why everyone couldn't just use the same prime, and, in fact, many applications tend to use standardized or hard-coded primes. But there was a very important detail that got lost in translation between the mathematicians and the practitioners: an adversary can perform a single enormous computation to "crack" a particular prime, then easily break any individual connection that uses that prime.

https://freedom-to-tinker.com/2015/10/14/how-is-nsa-breaking-so-much-crypto/

For the most common strength of Diffie-Hellman (1024 bits), it would cost a few hundred million dollars to build a machine, based on special purpose hardware, that would be able to crack one Diffie-Hellman prime every year.

NSA could afford such an investment. The 2013 "**black budget**" request, leaked as part of the Snowden cache, states that NSA has prioritized "investing in groundbreaking cryptanalytic capabilities to defeat adversarial cryptography and exploit internet traffic." It shows that the agency's budget is on the order of $10 billion a year, with over $1 billion dedicated to computer network exploitation, and several subprograms in the hundreds of millions a year.

TOP SECRET//SI/TK//NOFORN

(U) Investments

(U) Although the budget is declining, the mission is not. Prioritizing our requirements was a key element to produce a budget that meets customer needs, supports critical capabilities, addresses gaps, and helps to maintain a strategic advantage. In the FY 2013 NIP budget, the IC makes targeted investments in:

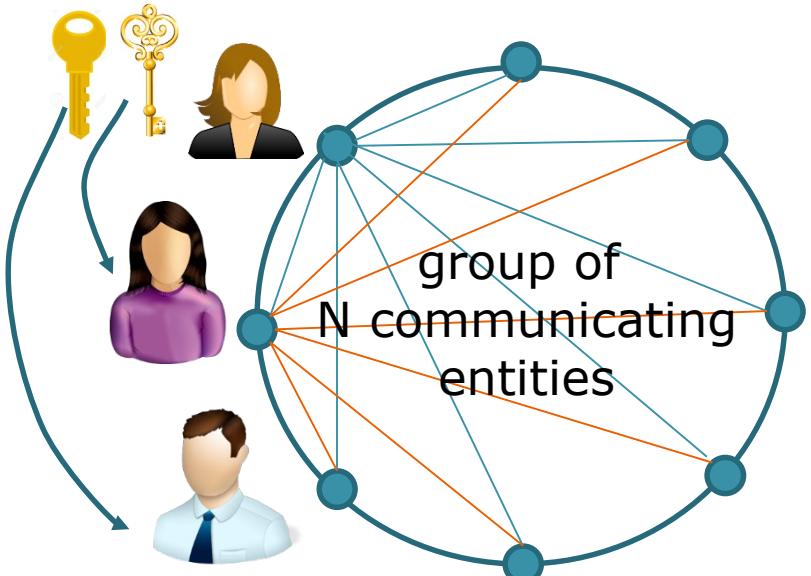
- (TS//SI//NF) ***Signals Intelligence (SIGINT).*** We are bolstering our support for clandestine SIGINT capabilities to collect against high priority targets, including foreign leadership targets. Also, we are investing in groundbreaking cryptanalytic capabilities to defeat adversarial cryptography and exploit internet traffic.

- (S//NF) ***Cybersecurity.*** As the cyber threat continues to grow, we sustain the budget for the Comprehensive National Cybersecurity Initiative and begin construction of a second High Performance Computing Center at Fort Meade, Maryland to keep pace with cyber processing demands.

Asymmetric Ciphers: D-H (cont.)

With DH algorithm
if n people were to <u>securely</u> communicate
$O(n^2)$ message would still
have to be exchanges.

No built-in mechanism to authenticate other users!!!



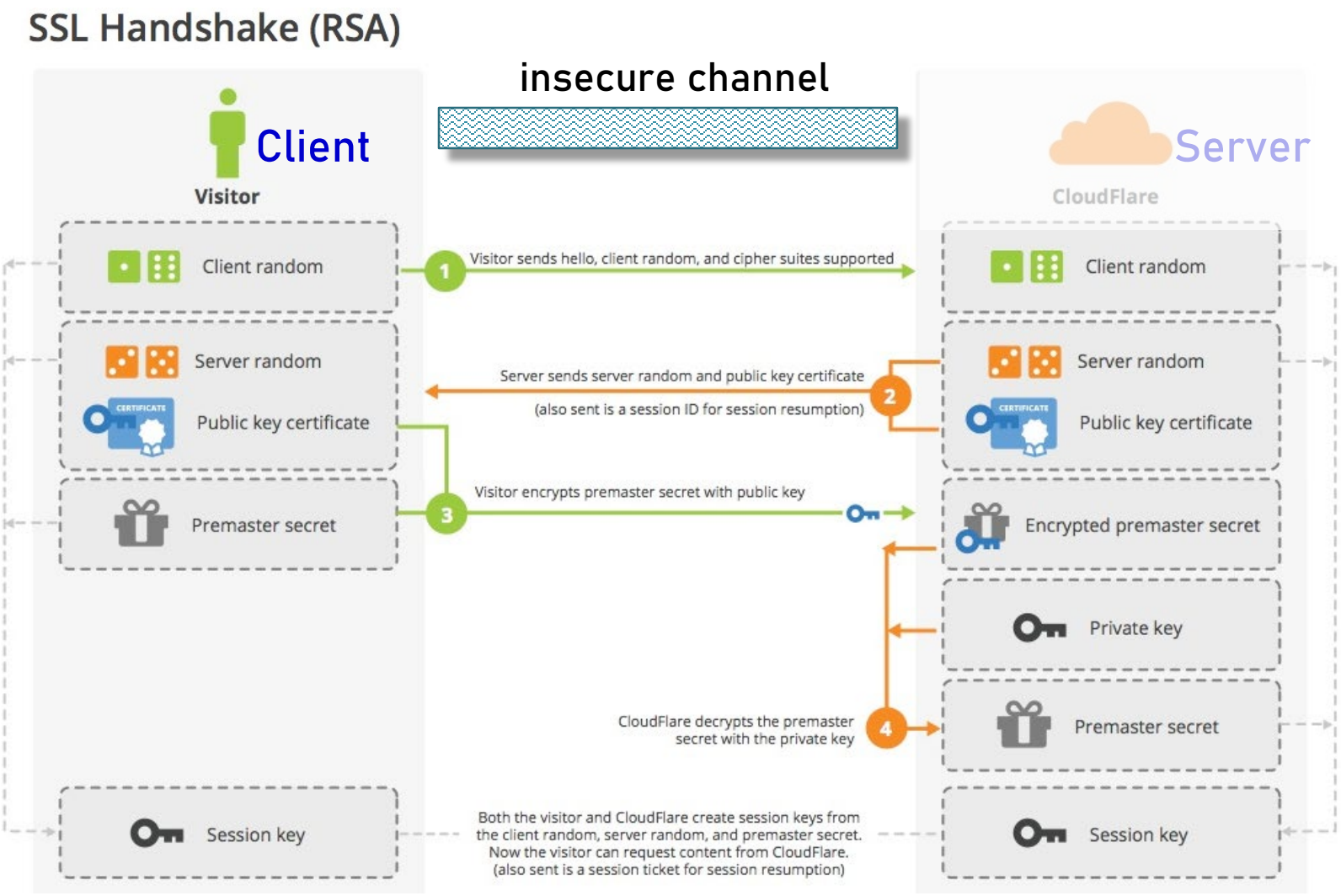
Asymmetric Ciphers: RSA

- **RSA** – Rivest, Shamir, Adleman (1978, MIT)
  - first practically deployable public-key algorithm for secure data transmission and other applications
  - was patented, but patent expired in 2000
  - RSA Security LLC – manufactures security solutions deploying RSA, was later sold to Dell …
    - spin-off company: VeriSign (1995), bought by Symantec and now DigiCert
  - based on practical difficulty of factoring the product of two large prime numbers
    - like DH uses modulus arithmetic, but in a different way

DH is used to generate a secret key [key agreement] …
RSA is used to exchange a secret key [key transport] …
for subsequent symmetric encryption.

Asymmetric Ciphers: RSA (cont.)

- internet protocols that use RSA: TSL, SSH, IPsec



Asymmetric Ciphers: RSA (cont.)

Example: **Excellent video!**

https://www.khanacademy.org/computing/computer-science/cryptography/modern-crypt/v/intro-to-rsa-encryption

Asymmetric Ciphers: RSA (cont.)

- **RSA** – basics of the math behind key establishment

(1) Choose two **random** large prime numbers **p** and **q**.
The larger the numbers, the more difficult it is to break RSA, but longer it also takes to perform encoding and decoding!!!
RSA Laboratories recommends that the product of **p** and **q** be 1024 bits long.

(2) Compute $n = p \cdot q$ and $z = (p-1) \cdot (q-1)$.

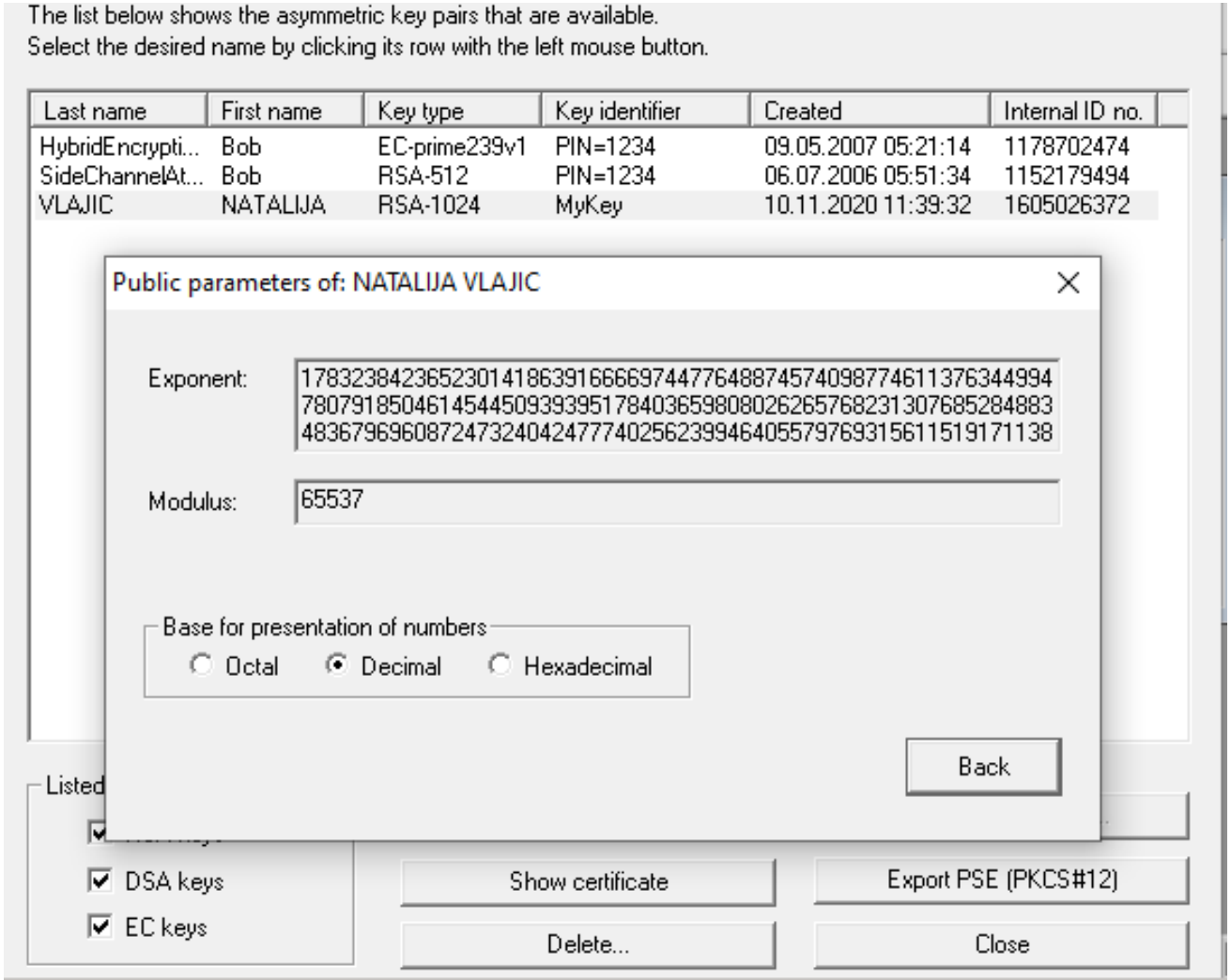
(3) **Choose** a number $e < n$ with no common factors with **z** other than 1. (e,n) – used in encryption, public key.

(4) Find a number **d** such that $ed - 1$ is exactly divisible by **z**.
That is, choose **d** such that $ed \bmod z = 1$.
(d,n) – used in decryption, private key.

(5) $K_{public} = (n, e)$, $K_{private} = (n, d)$

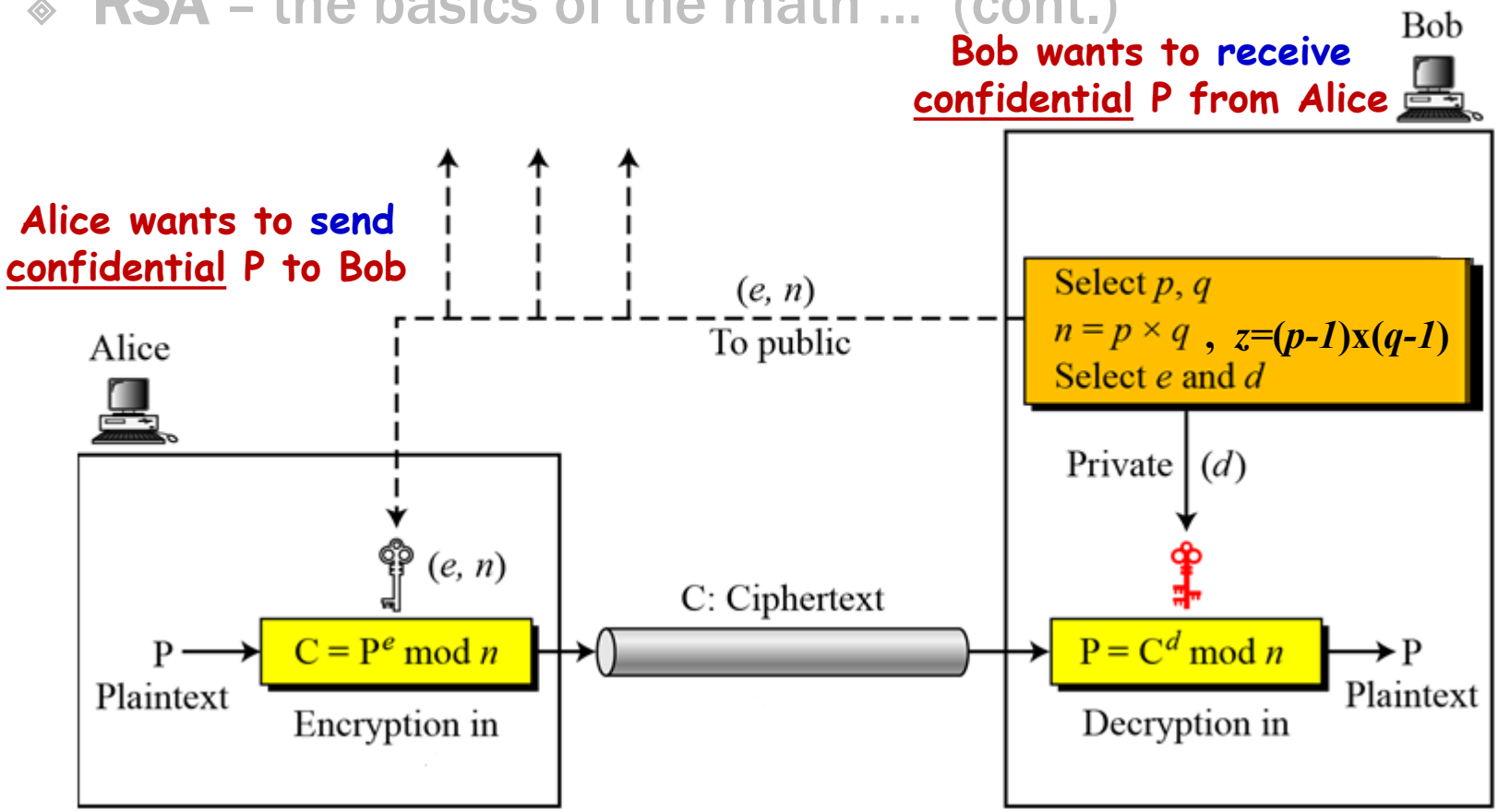
Asymmetric Ciphers: RSA (cont.)

Example: Lab 4 ...



Asymmetric Ciphers: RSA (cont.)

- RSA – the basics of the math … (cont.)



A Encrypts: $C = P^e \bmod n$

B Decrypts: $C^d \bmod n = (P^e \bmod n)^d \bmod n = P$

Asymmetric Ciphers: RSA (cont.)

- RSA – the basics of the math …
  - how can we prove:

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

below proof also holds if **e** and **d** are applied in reverse order

1) modulo rules allow:

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

2) theory of large prime numbers allows:

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

$$= P \quad - \quad \text{when } P<n$$

Has important practical significance: if plaintext is 'long', it cannot be encrypted at once – has to be broken into smaller parts/values!!!

Asymmetric Ciphers: RSA (cont.)

- **RSA** – important properties

1) Given $(e, n) = K_{public}$ it is/should be impossible to compute $(d, n) = K_{private}$.

provided **p** and **q** are properly randomized !!!

2) The public and private keys are 'commutative'.

$$K_{public}(K_{private}(P)) = K_{private}(K_{public}(P)) = P$$

$$K^{+}(K^{-}(P)) = K^{-}(K^{+}(P)) = P$$

Asymmetric Ciphers: RSA (cont.)

Example: RSA used to encrypt 8-bit messages

Bob chooses **p**=5, **q**=7. Then **n**=35, **z**=24.

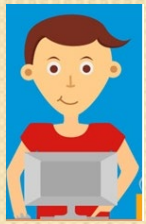
e=5 (so **e**, **z** relatively prime).

d=29 (so **ed**-1 exactly divisible by **z**).

Plaintext must be converted to a decimal number!!!

Encrypting 8-bit message: $0000\ 1100_2 = 12_{10}$.

	m	m^e	$c = m^e \bmod n$
Encrypt: (e,n)	12	24832	17



	c	c^d	$m = c^d \bmod n$
Decrypt: (d,n)	17	48196857210675091509141182522307169 7	12

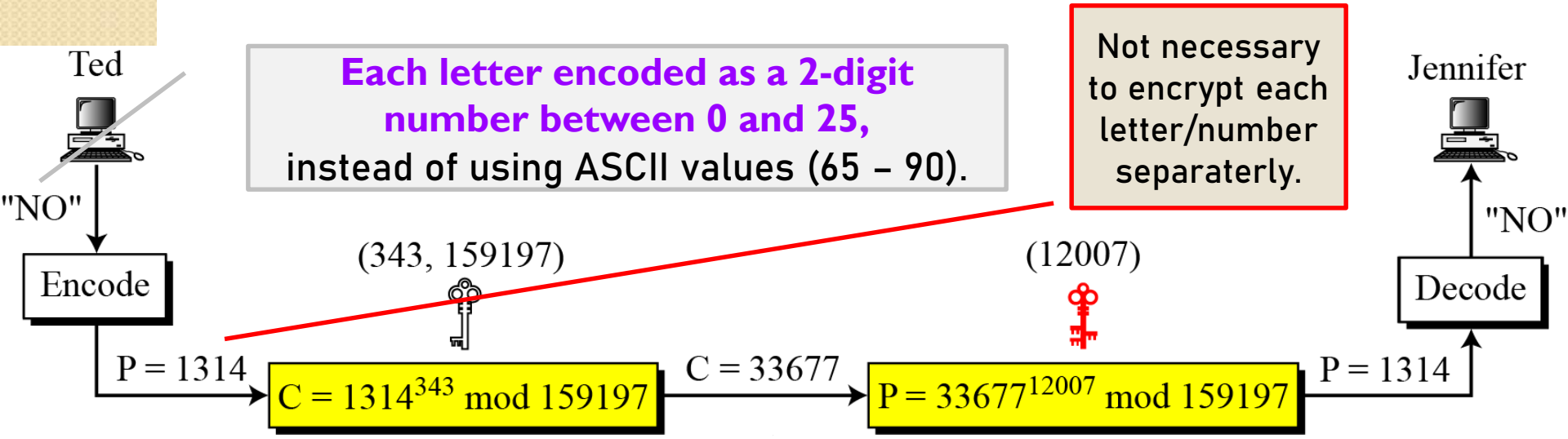
Asymmetric Ciphers: RSA (cont.)

Example: RSA used to encrypt letters

Jennifer creates a pair of keys for herself:
p=397 and **q**= 401 => **n**=159197 and **z**= 158400.

She then chooses **e**=343 and **d**=12007.

Show how Ted can send a 2-letter text message to Jennifer if he knows **e** and **n**.



Example: RSA and NSE Saga

Security firm RSA took millions from NSA: report

The National Security Agency paid $10 million to the security firm RSA to implement intentionally flawed encryption, according to a new report.

From 2004 to 2013, RSA shipped security software — BSAFE toolkit and Data Protection Manager — that included a default cryptographically secure pseudorandom number generator, Dual EC DRBG that was later suspected to contain an alleged secret National Security Agency backdoor.

In 2014, the Snowden leaks revealed how the NSA was effectively infiltrating crypto standards efforts to take control of them and make sure that backdoors or other weaknesses were installed.

On 20 December 2013, Reuters' Joseph Menn reported that NSA secretly paid RSA Security $10 million in 2004 to set Dual_EC_DRBG as the default CSPRNG in BSAFE. The story quoted former RSA Security employees as saying that "no alarms were raised because the deal was handled by business leaders rather than pure technologists".

https://en.wikipedia.org/wiki/RSA_Security

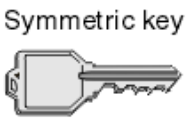
https://www.reuters.com/article/us-usa-security-rsa/exclusive-secret-contract-tied-nsa-and-security-industry-pioneer-idUSBRE9BJ1C220131220

https://www.techdirt.com/articles/20131220/14143625655/nsa-gave-rsa-10-million-to-promote-crypto-it-had-purposely-weakened.shtml

http://news.cnet.com/8301-1009_3-57616205-83/security-firm-rsa-took-millions-from-nsa-report/

AES vs RSA vs DH

	AES	DH	RSA
symmetric/ asymmetric	symmetric	asymmetric	asymmetric
uses	**encrypt data** (symmetric key must be exchanged)	**generate symmetric key** (problematic over insecure channel)	• encrypt data (slow) • **exchange symmetric key** • **prove sender's authenticity** (public keys must be known/exchanged)



RSA vs DH for key exchange

Example: RSA to exchange a symmetric key – Scenario 3

Let's use symmetric encryption.
Send me your public key inside your **Digital Certificate** !!!
I'll generate the key and send it back to you

B_+ B_-

Ok. Here is my Digital Certificate.

S_+ S_-

S_+

K

$S_+(K)$

Assuming the Digit. Certificate is 'vetted' and signed by a trusted 3rd party, Bob now can be sure that the message really came from Server !!!

data encrypted using K

How does this compare to DH ???

RSA Keys in Real World ...

The security available with a 1024-bit key using asymmetric RSA is considered approximately equal in security to an 80-bit key in a symmetric algorithm.

2048-bit RSA keys are equivalent to 112-bit symmetric keys.

2048-bit keys are sufficient until 2030.

As of 2020 the largest RSA key publicly known to be cracked is RSA-250 with 829 bits.

https://en.wikipedia.org/wiki/Key_size

Security Researchers Able to Crack 1024-bit RSA Encryption

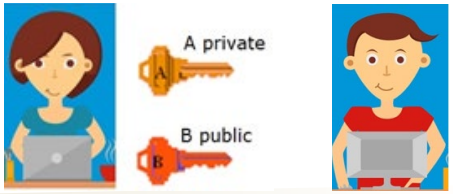
By **Angad Singh** - 05/07/2017

Security researchers discovered a critical crash (CVE-2017-7526 called) in the GnuPG cryptographic library that allowed researchers to completely break RSA 1024-bit RSA encryption and successfully get the secret key to decrypt data. Therefore, Security researchers able to crack 1024-bit RSA encryption.

GnuPG is a hybrid encryption software that uses a combination of traditional symmetric key encryption for public key speed and encryption to facilitate secure key exchange, usually using the recipient's public key to encrypt a session key that it uses once.

https://www.officialhacker.com/security-researchers-able-to-crack-1024-bit-rsa-encryption/

RSA Applications



Application of RSA Cryptography

- **protect. of <u>data confidentiality</u> & <u>user/message authenticity</u>**
  (symmetric key confid.)

- other possible more common uses:

  a) **digital envelopes** = **fast exchange** of confidential messages (secret message & secret key sent at once)

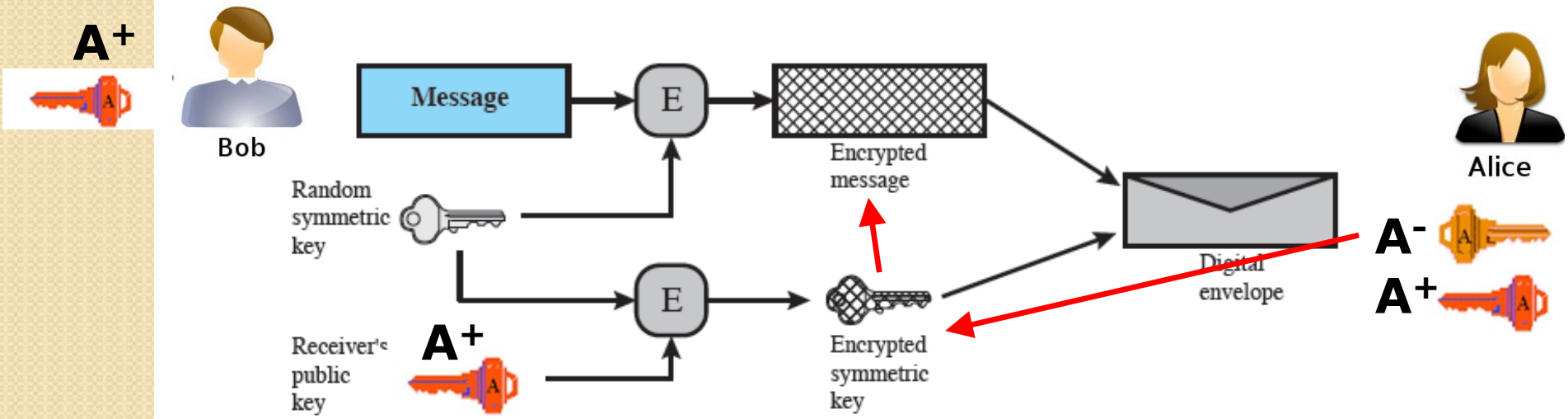
  b) **digital signature** =
  = **message integrity** + **message authentication**, where

  message integrity – guarantees that the message has not been changed

  message authentication – authenticates the sender of the message

RSA Applications (cont.)

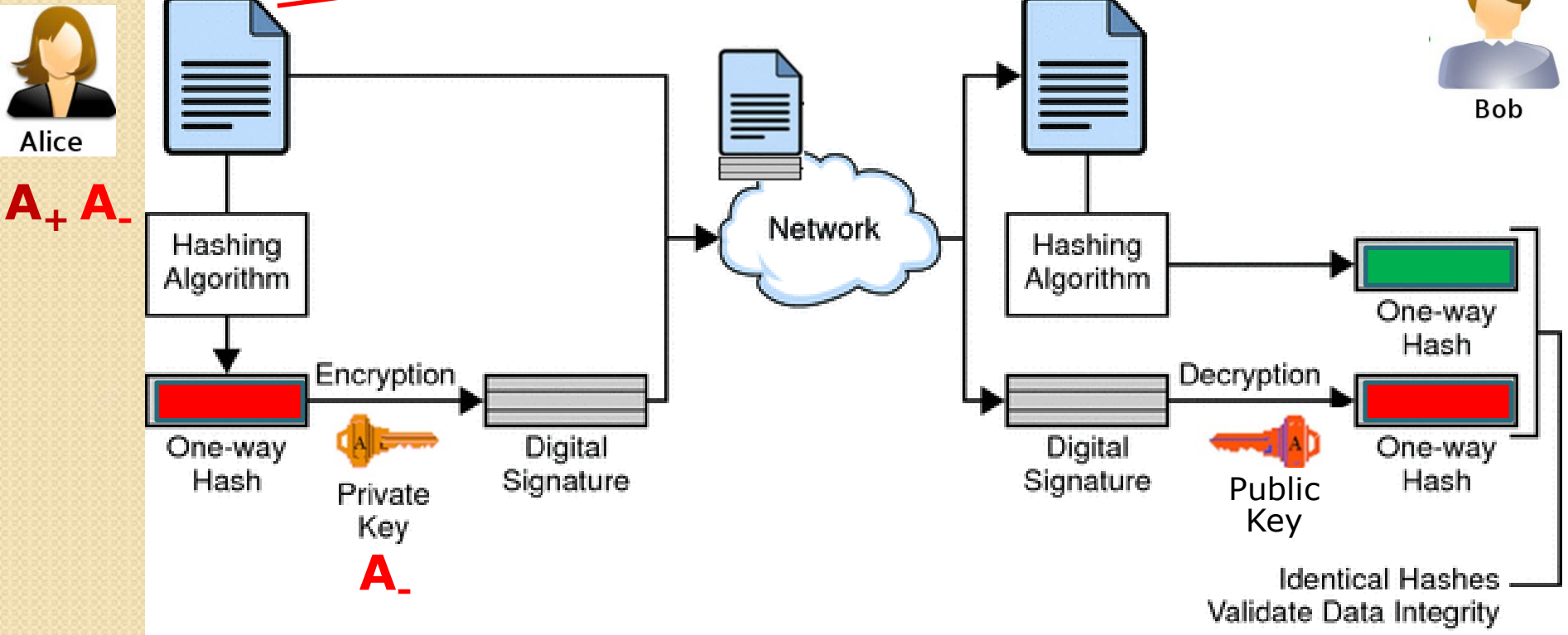
- **Digital Envelope** – use of asymmetric encryption for fast exchange of confidential messages
  1) generate random symmetric key $K_{symmetric}$
  2) encrypt message using $K_{symmetric}$ – digital letter
  3) encrypt $K_{symmetric}$ using receiver's public key K^+ - protective digital envelope
  4) send the two together !!!



RSA Applications (cont.)

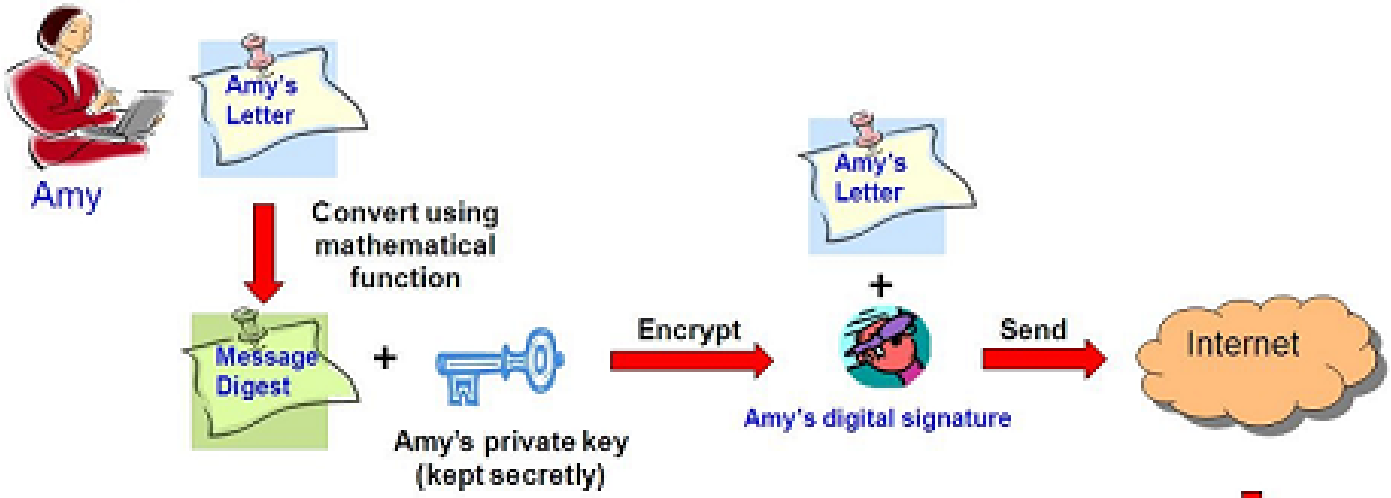
In some cases the confidentiality is not required - data sent in plaintext) but we want to be able to ensure …

- **Digital Signature** - use of asymmetric encryption to protect message integrity + sender authenticity

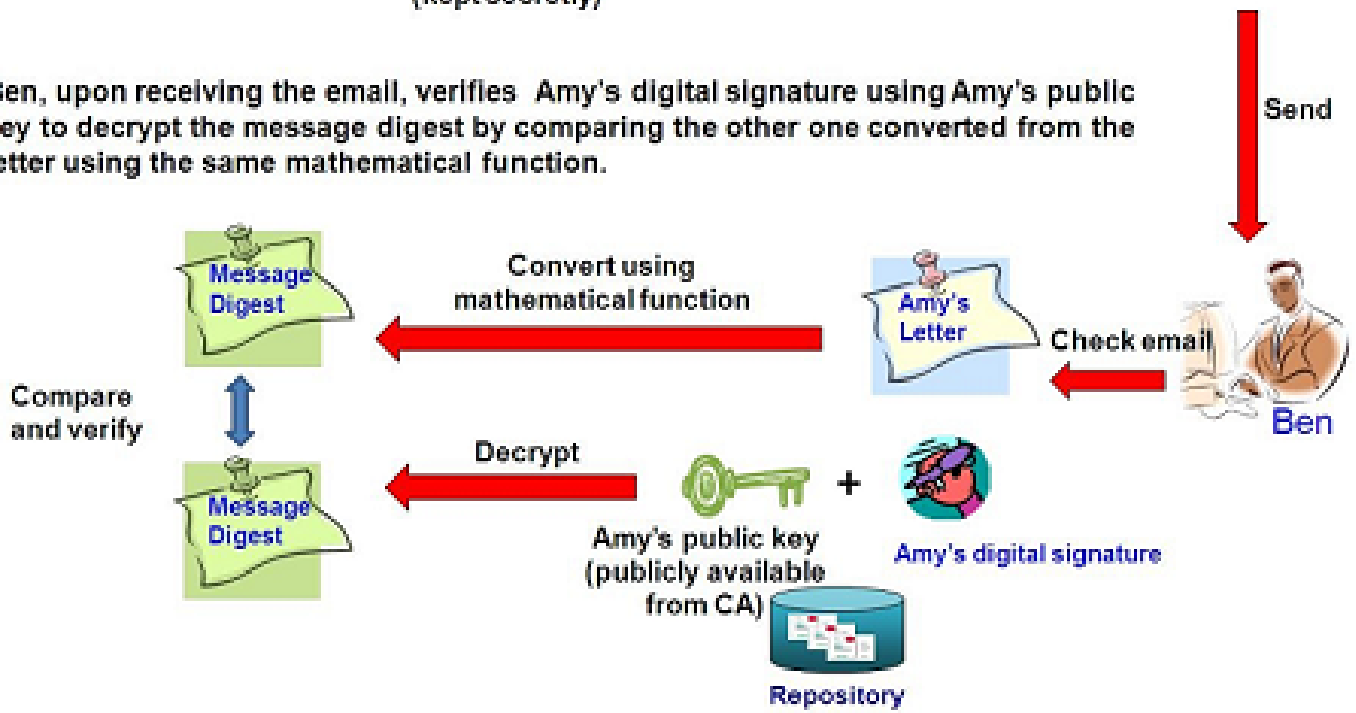


Digital Signature

1. Amy converts her letter into a message digest by using a mathematical function. She then creates her digital signature by encrypting the message digest using her private key. Her letter, together with her digital signature are sent to Ben via email.



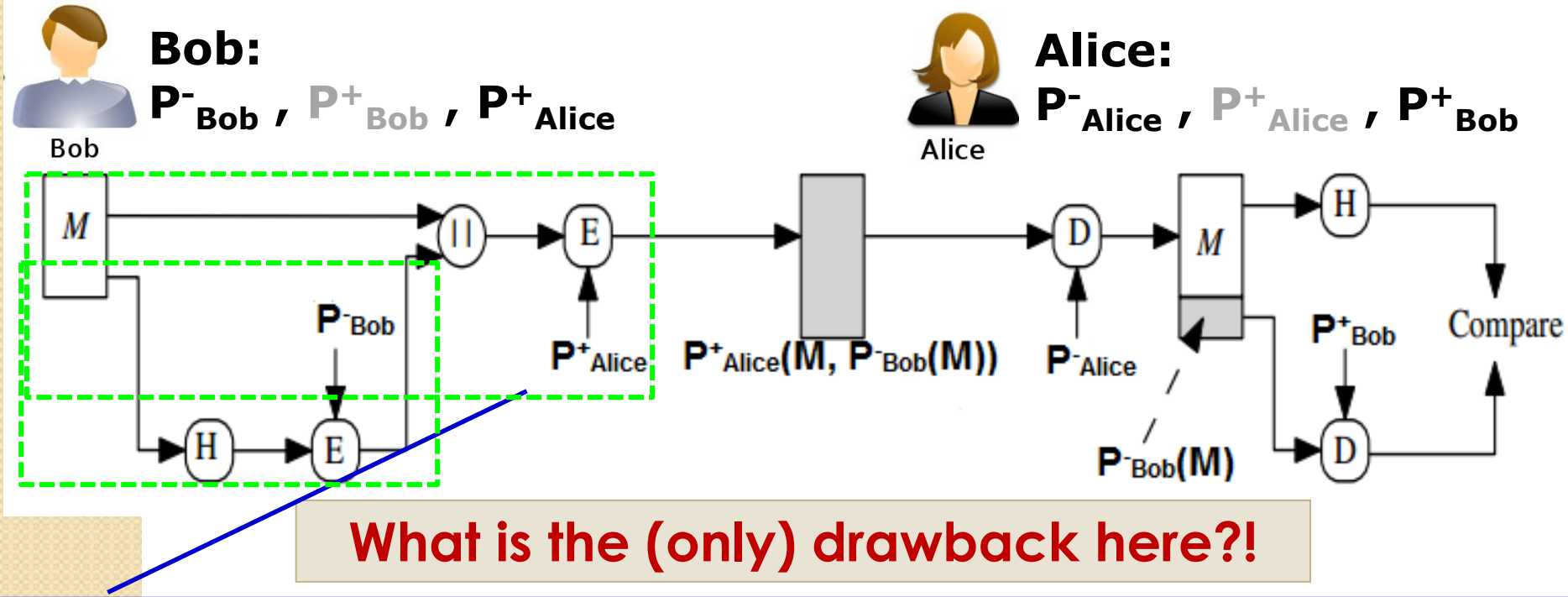
2. Ben, upon receiving the email, verifies Amy's digital signature using Amy's public key to decrypt the message digest by comparing the other one converted from the letter using the same mathematical function.



RSA Application (cont.)

Example: Public encryption for all three – message integrity, authentication and confidentiality

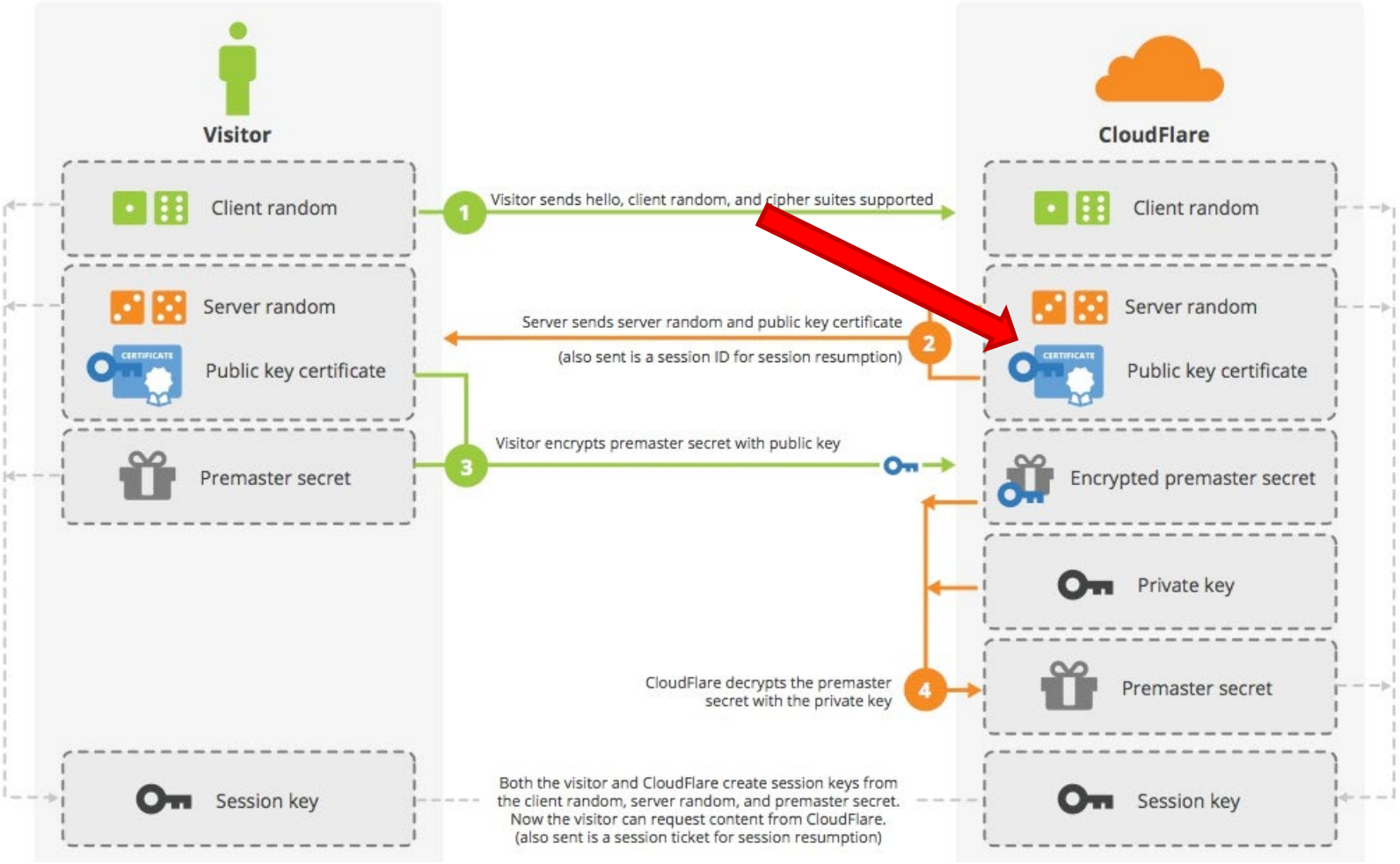
(digital signatures + confidentiality)



What is the (only) drawback here?!

NOTE: this is theoretically OK, but practically very slow. A better solution would be for Bob to generate a symmetric key K, use K to encrypt the message & digest, and send K encrypted using Alice's public key …

Public-Key / Digital Certificates



Public-Key / Digital Certificates (cont.)

- **Reliable Public-Key Distribution** – must involve a trusted third party
  - **Certificate Authority** – a trusted government agency or a for-profit institution that issues Digital Certificates
    - IdenTrust, DigiCert, GlobalSign, …
  - **Digital Certificate** – digital document that binds a public key to an identity (person or organization) and contains:

Serial Number: Used to uniquely identify the certificate.

Subject: The person, or entity identified.

Signature Algorithm: The algorithm used to create the signature.

Signature: The actual signature to verify that it came from the issuer.

Issuer: The entity that verified the information and issued the certificate.

Valid-From: The date the certificate is first valid from.

Valid-To: The expiration date.

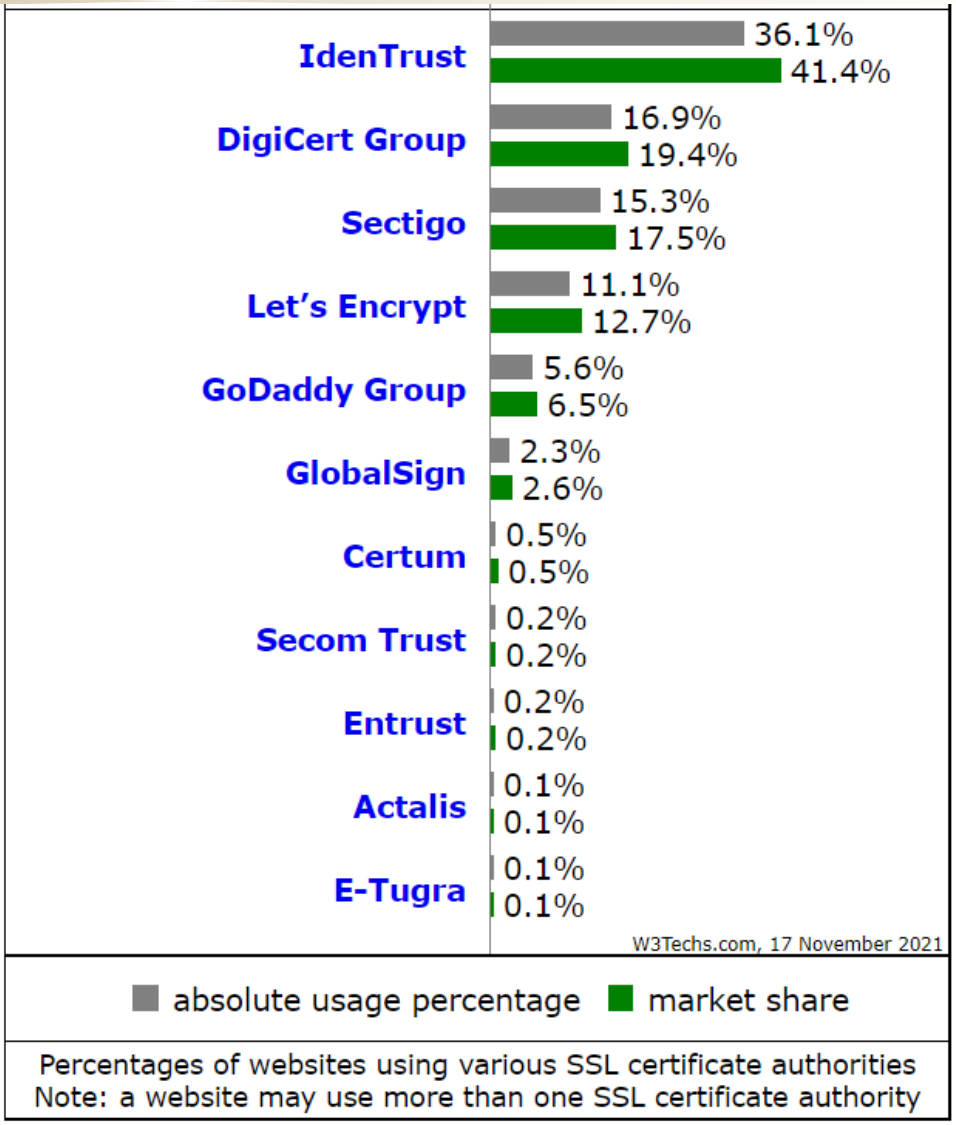
Key-Usage: Purpose of the public key (e.g. encipherment, signature, certificate signing...).

Public Key: The public key.

Public-Key / Digital Certificates (cont.)

Example:

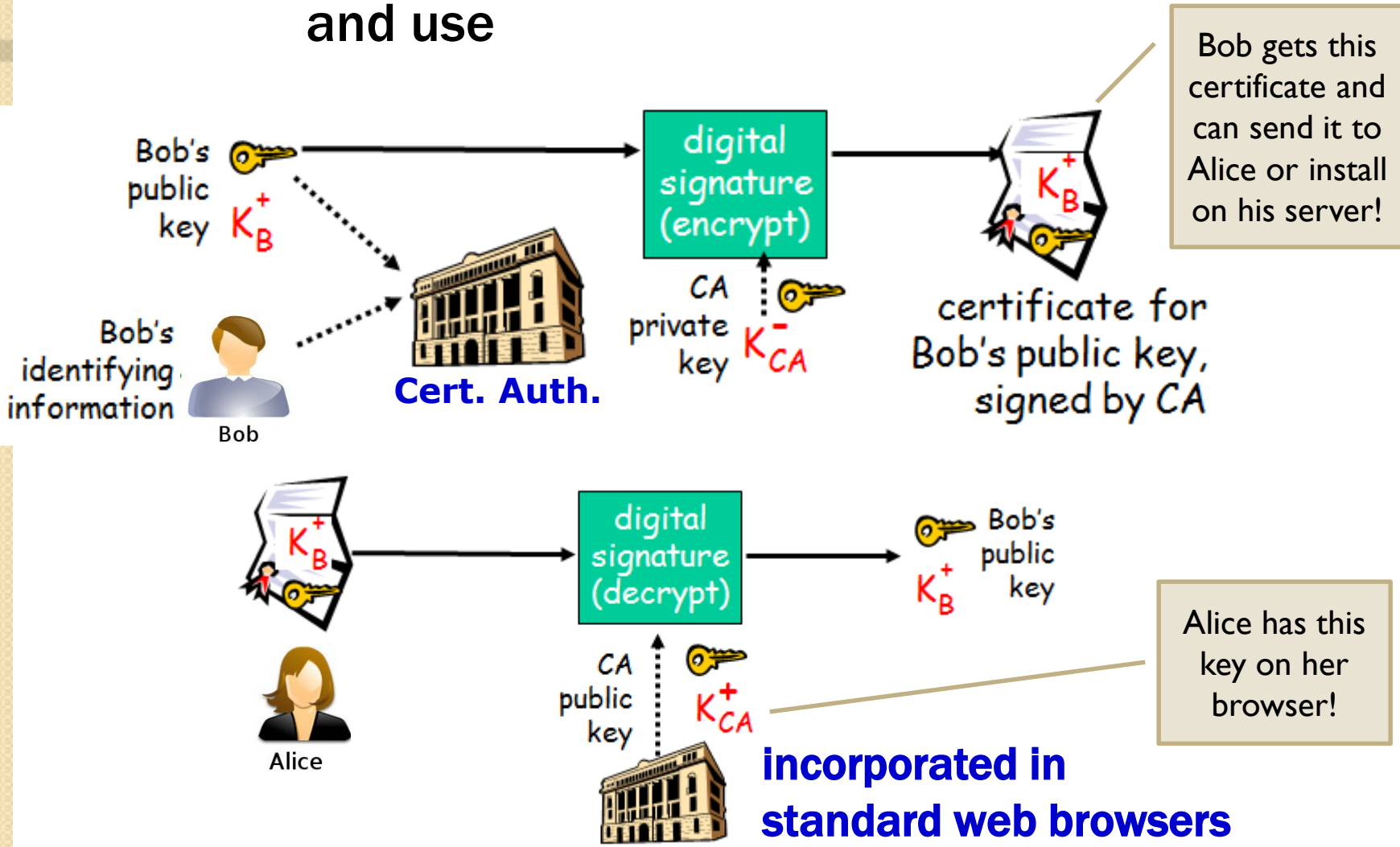
CA Prevalence



https://w3techs.com/technologies/overview/ssl_certificate

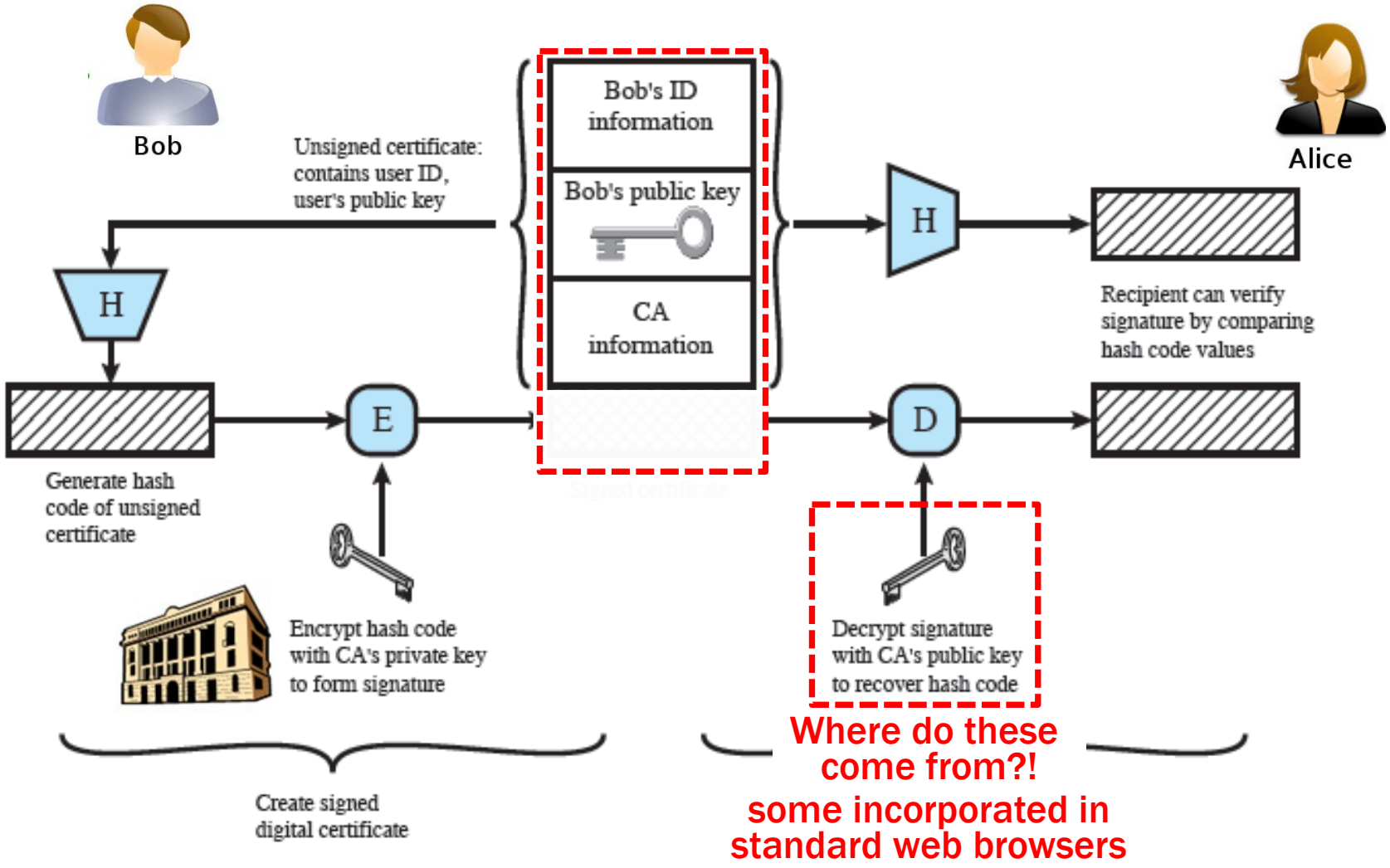
Public-Key / Digital Certificates (cont.)

Example: Creation of public-key certificate: creation and use



Public-Key / Digital Certificates (cont.)

Example: Creation & verification of a digital certificate



Encoding vs. Encryption vs. Hashing

- **Message Encoding vs. Encryption vs. Crypto. Hashing**
  - all three transform message into another 'format'
  - encoding and encryption are reversible, hashing is not!

1) **message encoding** – transforms data to another format so that it can be properly/safely consumed by a different type of system

- does not aim to keep information secret
- does not require a key
- encoding scheme is publicly available and relatively simple/fast to perform

000d	00h		(nul)	016d	10h	▶	(dle)
001d	01h	☺	(soh)	017d	11h	◀	(dc1)
002d	02h	☻	(stx)	018d	12h	↕	(dc2)
003d	03h	♥	(etx)	019d	13h	‼	(dc3)
004d	04h	♦	(eot)	020d	14h	¶	(dc4)
005d	05h	♣	(enq)	021d	15h	§	(nak)
006d	06h	♠	(ack)	022d	16h	▬	(syn)
007d	07h	•	(bel)	023d	17h	↨	(etb)
008d	08h	◘	(bs)	024d	18h	↑	(can)
009d	09h		(tab)	025d	19h	↓	(em)
010d	0Ah		(lf)	026d	1Ah		(eof)
011d	0Bh	♂	(vt)	027d	1Bh	←	(esc)
012d	0Ch	♀	(np)	028d	1Ch	∟	(fs)
013d	0Dh		(cr)	029d	1Dh	↔	(gs)
014d	0Eh	♫	(so)	030d	1Eh	▲	(rs)
015d	0Fh	☼	(si)	031d	1Fh	▼	(us)

special characters on a web page

Encoding vs. Encryption vs. Hashing (cont.)

Message Encoding vs. Encryption vs. Hashing (cont.)

2) **message encryption** – transforms data to another format that <u>cannot</u> be easily consumed by anybody but the intended recipient(s)

- aims to keep information secret
- requires a key
- encryption scheme is publicly available but quite complex to perform/break

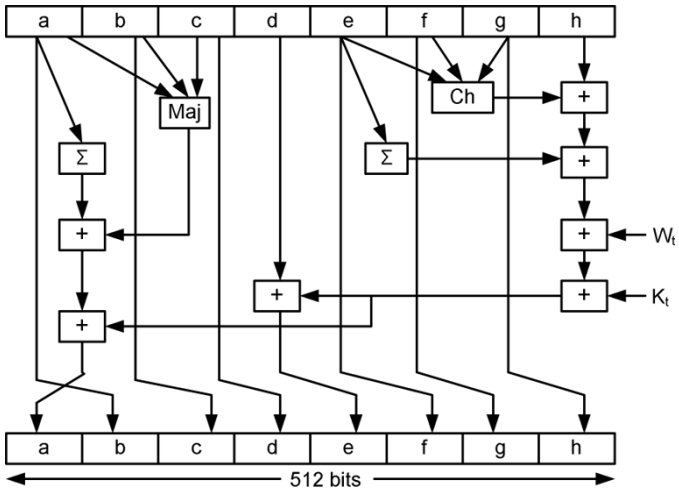
```
-----BEGIN PGP MESSAGE-----
Version: GnuPG v1.4.5 (GNU/Linux)

hQIOAOuHn1ue4n32EAf/UEF6JLrap1OBMdKMvb+Dz9GvoijUixH+gbcpi9qGa+43
vC3ktMwo7OWqPyJseVRSPBOv6dOwy65KrzrHwhOHO/CKEk2O5STAwzj6C3USgDfZ
6E+Gc4iumMl725JNahJzcL5ED33LFdZ6uoEjgqggxG1dFwvwksRHA4+VU9Bcd5eL
T9aRVbkXNxXkQn2FWhWuhPQFNWLwIVrDd9TPtDvpRTl6YiB1AM9ks3HlYZHL7mfR
Hk9yfy1nGXdhiO6EDvvTvd/Lq1xsFjKh6y/pG6NxABGdT6VoeWGVtQGqwpbOZGgq
xoSYkWm8MmAkkqYXZLraSEzyxxxu4cQzvzz3vrpN3AgAhObP2eUFU29EJAQpdKJW
fKAhohPVpd6+ETnzL53VLg1IJJdNG1pIziO9alNnYmDSnt2EwAELqTUl3jPiGYt5
cvSUBe3ER4/CkjvYXOVaO7ezHmCAkQpB2ILV8OwI74DQn7tNKf2gJnwzkYAF7yyf
XFG1J8oaLpRV499mN71Nfo+ZV2HrR9xti+jUPFv+H+ROt4fMmAU5I95UksQFe/A9
YUdSBAEqKkW9zLDgpWS2oxJymGufBdhzxpw7uJlzrwsHIYIt7PSeJG4VO+xJqHvO
1qHXSukK648F1OImmVUM9csPOcvfOMZeAgh4i+HYQvFF/kGHp6ogevD4pVhztbzd
F9JhAbJSeOvZKZFPhzjgX+mCgvzVRniSdDg7wc3+YKNei2zQrmTsiiO6JyhQV2OI
tAqTk572zdZbrCtSgcthrN/uxbJSNnw4X9IZbWtFOUr3lr676II8Q112ttO3IVCe
fF/pZA==
=sPWf
-----END PGP MESSAGE-----
```

Encoding vs. Encryption vs. Hashing (cont.)

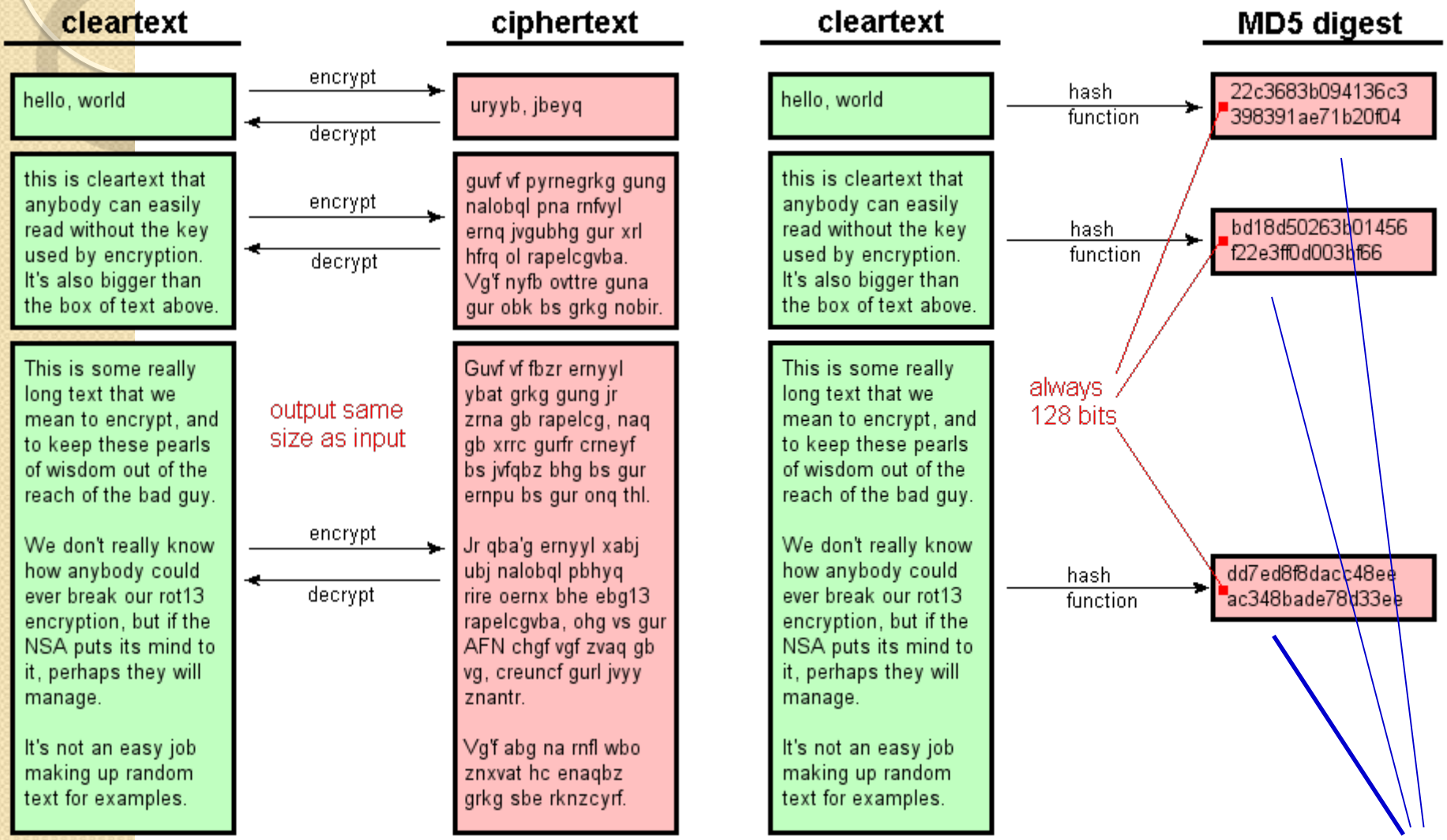
- Message Encoding vs. Encryption vs. Hashing (cont.)

3) **message hashing** – used to **validate the integrity** of a given content by producing a **fixed-length** string with following attributes:

- does not require a key
- hashing algorithms are publically available
- the same input will always produce the same output
- any modification to the input should result in a drastic change to the output



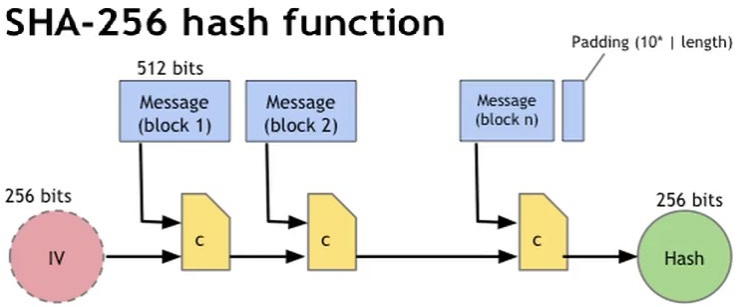
Encoding vs. Encryption vs. Hashing (cont.)



In case of hashing, the output 'size' is always constant / does not depend on the size of the input !!!

Hashing

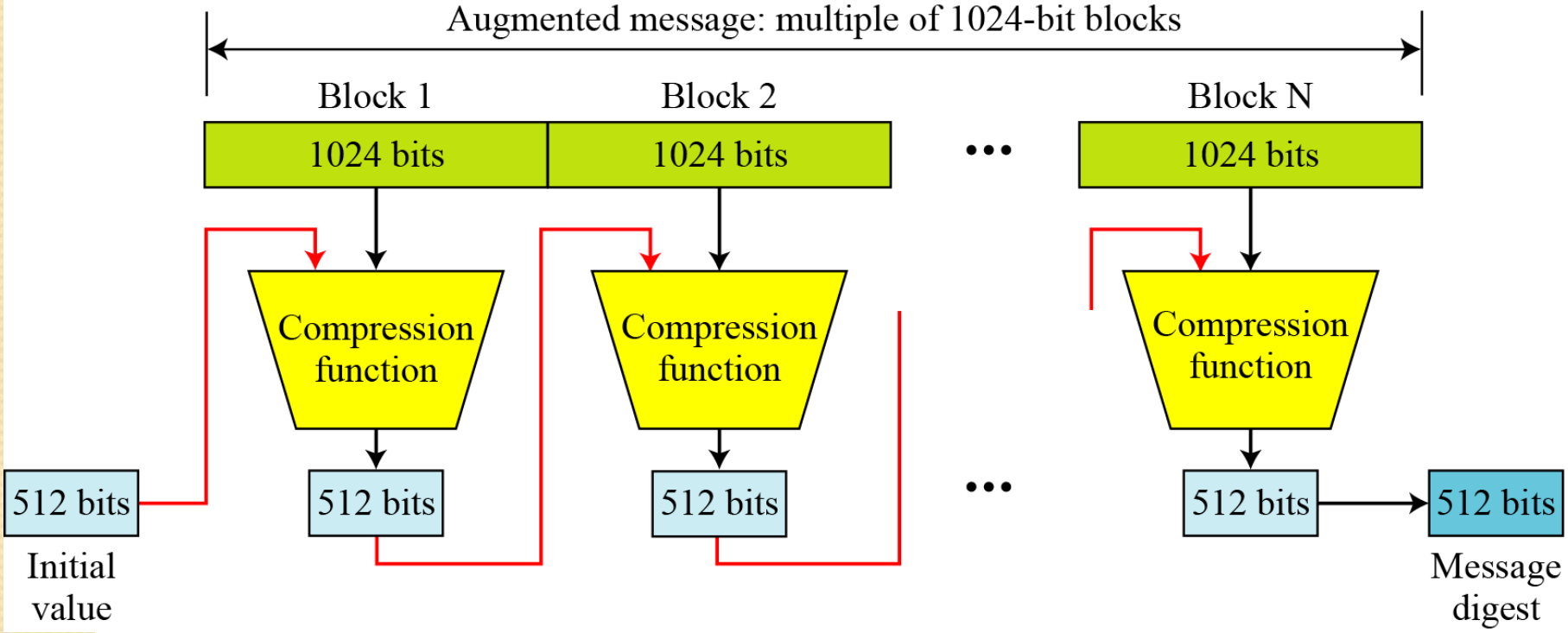
- **Message Integrity** – accomplished through the use of cryptographic hash functions
  - hash function creates a small fixed-size digital 'summary' of the message that can be used as a message fingerprint, aka hash or message digest
  - typical hash size: 128, 160, 256, 512 bits
  - popular standards:
    (a) Message Digest 5 (MD5) – no longer secure
    (b) Secure Hash Algorithm (SHA-2: SHA 256 & SHA 512)



https://en.wikipedia.org/wiki/Secure_Hash_Algorithms

Hashing (cont.)

Example: Message digest creation with SHA-512



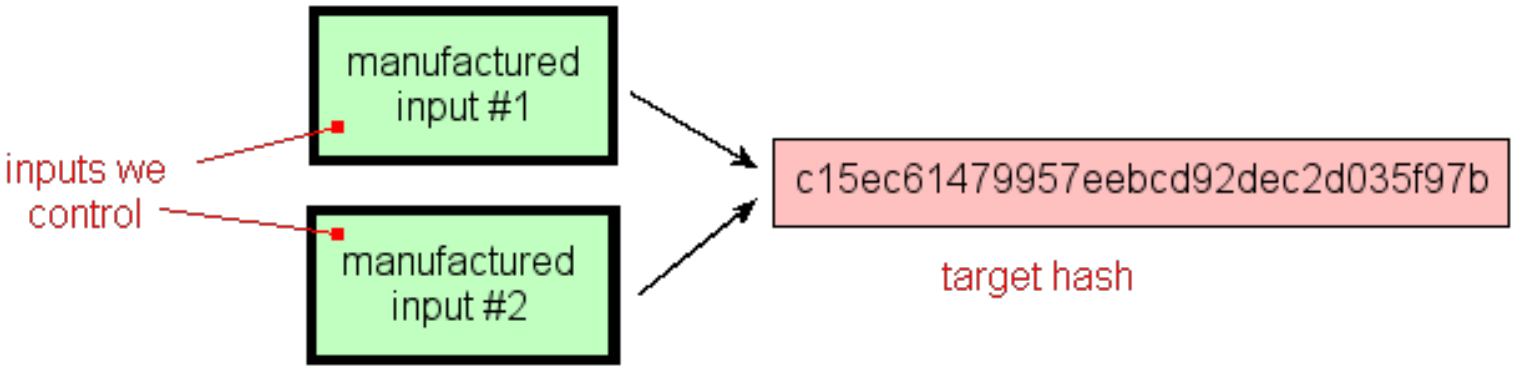

```
SHA512/256("The quick brown fox jumps over the lazy dog")
0x dd9d67b371519c339ed8dbd25af90e976a1eeefd4ad3d889005e532fc5bef04d

SHA512/256("The quick brown fox jumps over the lazy dog.")
0x 1546741840f8a492b959d9b8b2344b9b0eb51b004bba35c0aebaac86d45264c3
```

Hashing (cont.)

- **Hash Function Criteria** – to be eligible for a hash a function needs to meet 6 important criteria:
  - Hash function h can be applied to block of data of any size.
  - Hash function h produces a fixed-length output.
  - h(M) is relatively easy to compute for any given M, making both hardware and software implementation practical.
  - **Collision Resistance**.
  - **Preimage Resistance**.
  - **Second Preimage Resistance**.

Hashing (cont.)

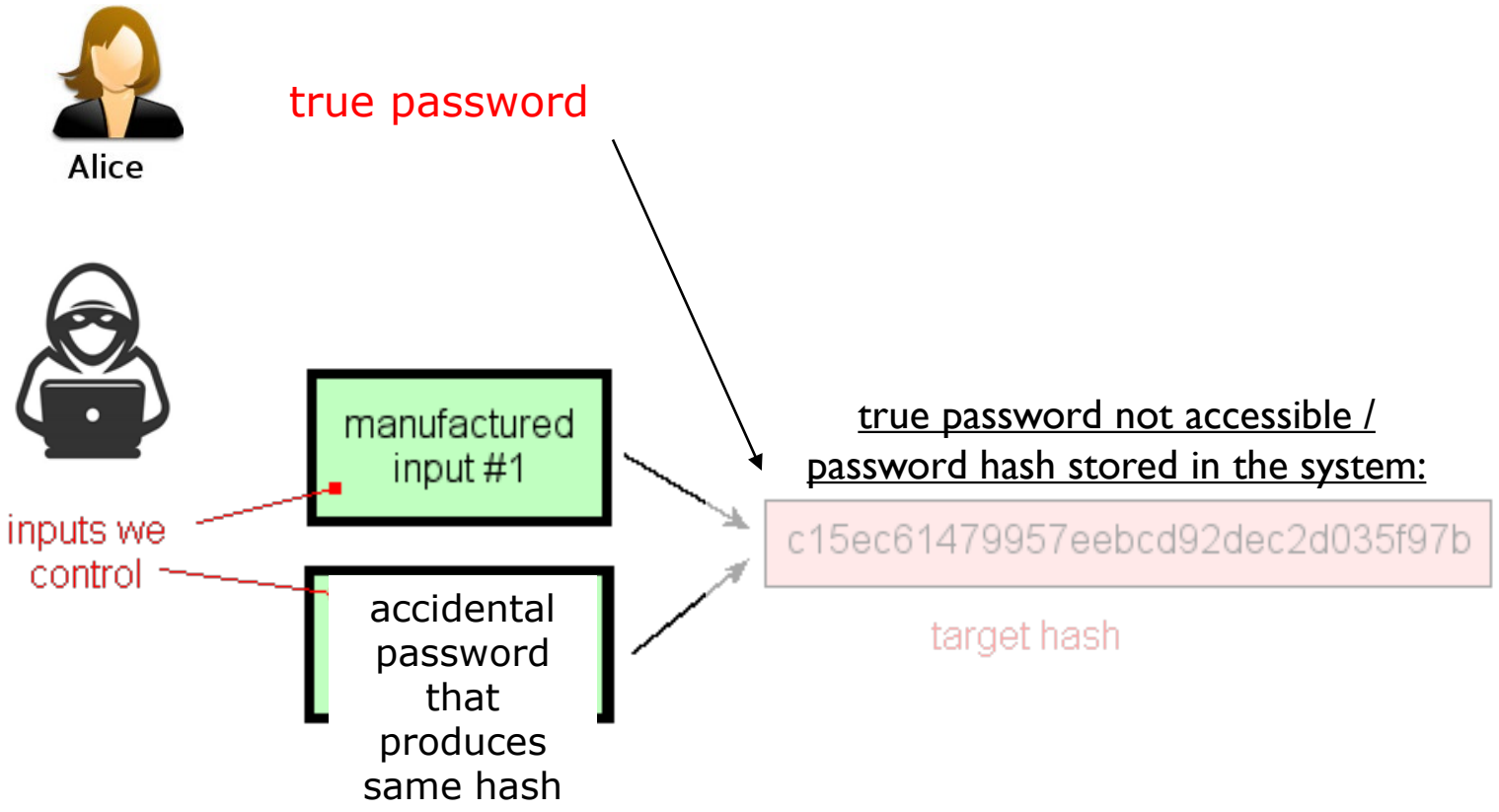
- Hash Function Criteria (cont.):
  - *collision: two messages create the same digest*
  - **Collision Resistance** or **Strong Collision Resistance**: must be *extremely difficult* to find **any two** M and M' such that h(M) = h(M')
  - **if strong collision is possible => digital signatures become meaningless**
  - *also relevant to online password cracking*



Hashing (cont.)

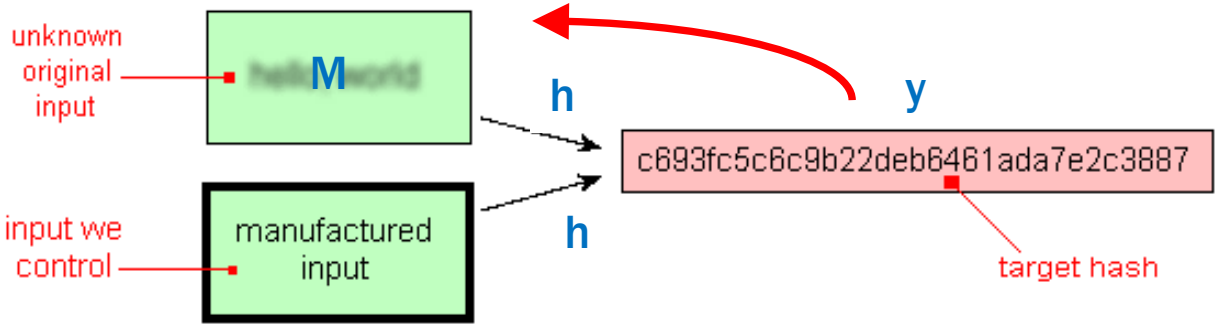
Example: Strong Collision Resistance example

(online password cracking)



Message Integrity (cont.)

- Hash Function Criteria (cont.):
  - **Preimage Resistance** or **One Wayness**: given a hash function h and y=h(M), it must be extremely difficult for Eve to find any message M' such that y=h(M')
  - we should not be able to work 'backwards' and (re)create the original message from a given hash
  - *relevant for off-line password cracking*

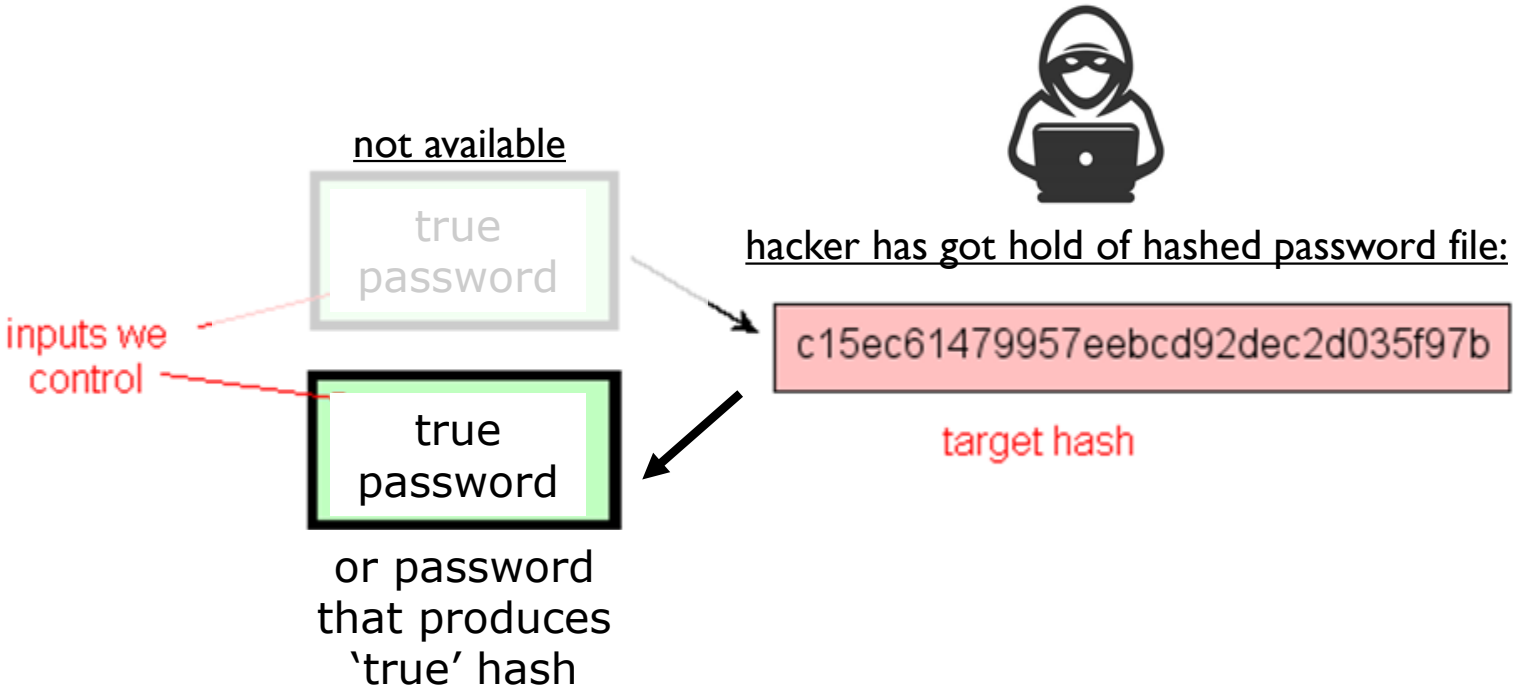


http://www.unixwiz.net/techtips/iguide-crypto-hashes.html

Message Integrity (cont.)

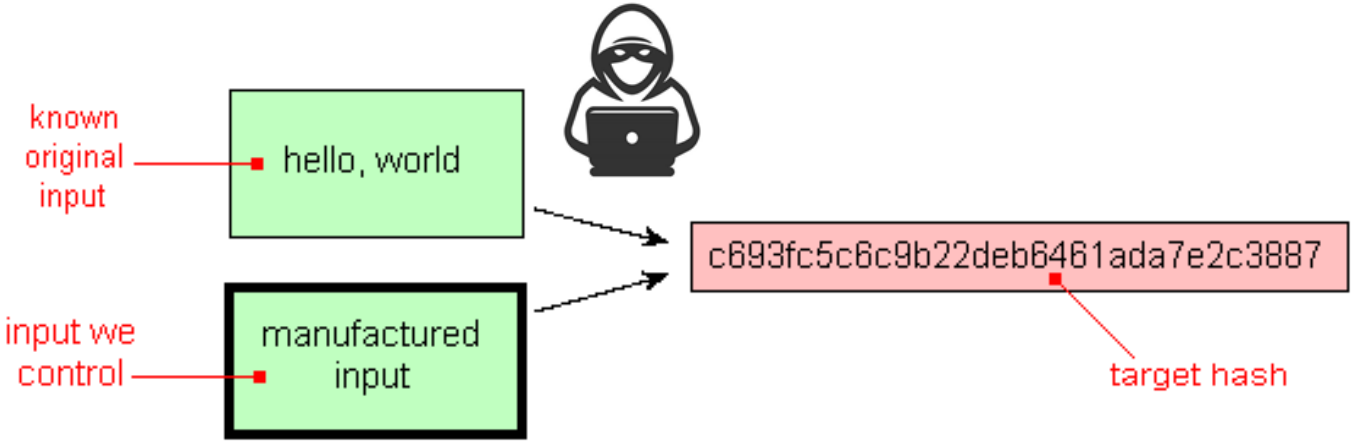
Example: Preimage Resistance example

(off-line password cracking)



Hashing (cont.)

- Hash Function Criteria (cont.):
  - **Second Preimage Resistance** or **Weak Collision Resistance**: given M and its hash h(M) it should be extremely difficult for Eve to find a second/another message M' such that h(M)=h(M')
  - property intended to prevent an adversary from appending a falsified message to a given hash



http://www.unixwiz.net/techtips/iguide-crypto-hashes.html

Hashing (cont.)

Example: **Second Preimage Resistance example**

(alter the content of a 'signed' message)