

SECURITY



DR. RAMZI SAEFAN

DONE BY
YARA AL-JARZI

 POWERUNIT 

COMPUTER AND NETWORK SECURITY

Jonathan Katz

Modified by: Dr. Ramzi Saifan



“SECURITY”

- Most of computer science is concerned with *achieving desired behavior*
- Security is concerned with preventing undesired behavior
 - Different way of thinking!
 - An enemy/opponent/hacker/adversary who is actively and maliciously trying to circumvent any protective measures you put in place



ONE ILLUSTRATION OF THE DIFFERENCE

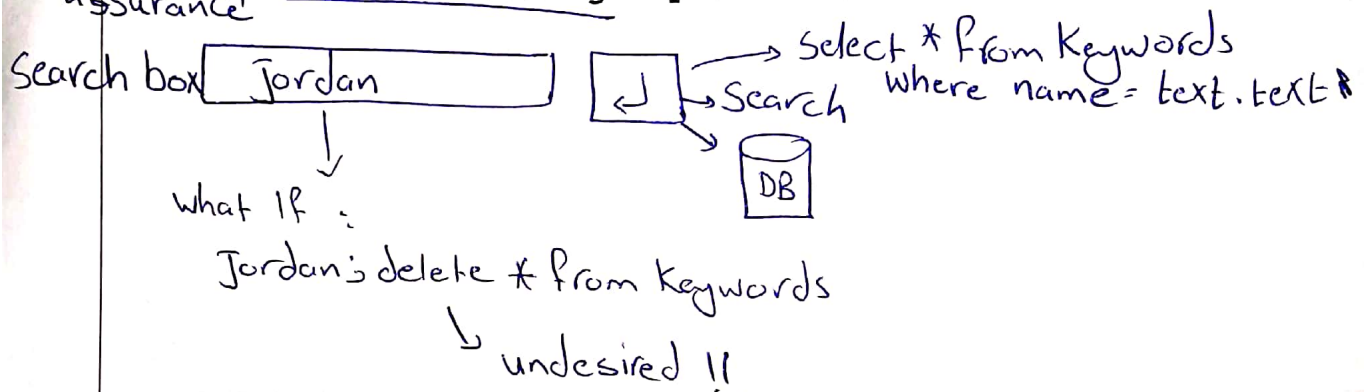
- Software testing determines whether a given program implements a desired functionality

- Test I/O characteristics
- Q/A

- How do you test whether a program does *not* allow for *undesired* functionality?

①
quality
assurance

- Penetration testing helps, but only up to a point



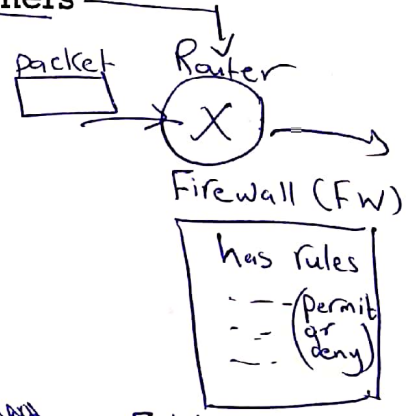
SECURITY IS INTERDISCIPLINARY

- Draws on all areas of CS
 - Theory (especially *cryptology*)
 - Networking
 - Operating systems
 - Databases
 - AI/learning theory
 - Computer architecture/hardware
 - Programming languages/compiler
 - HCI, psychology

FORTUNATELY, WE ARE WINNING THE SECURITY BATTLE

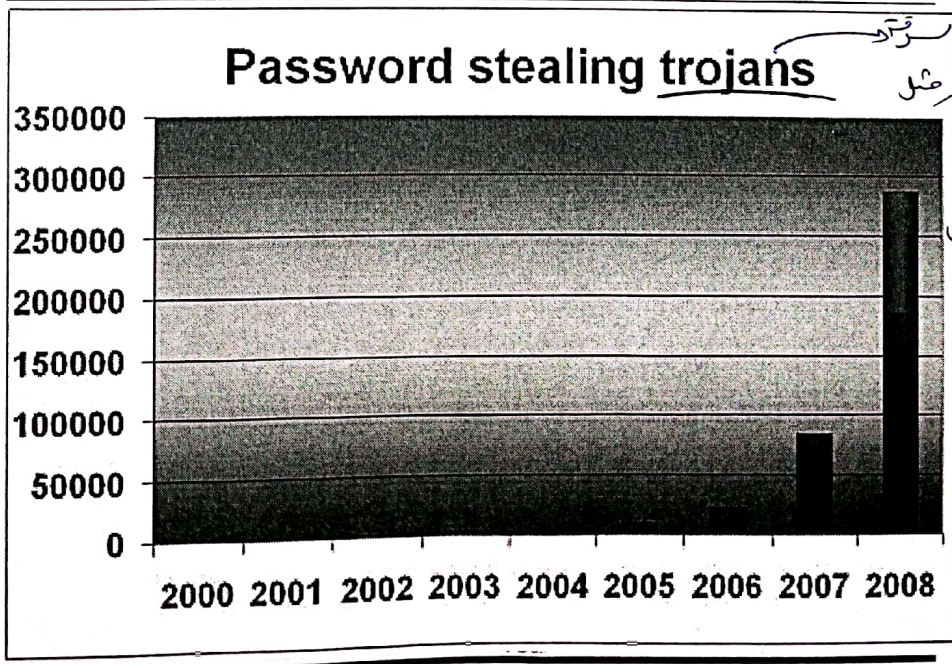
- Strong ^{علم التشفير} cryptography
- Firewalls, intrusion detection, virus scanners
- Buffer overflow detection/prevention
- User education

Intrusion detection system (IDS)
 Intrusion prevention system (IPS)



ex: ~~FW~~ FW test rules on the packet
 → one permit it passes without
 Continue other rules.

REALLY??!



اسلوب سرقة
 الباسورد
 صورته اد
 تطبيق
 كتوي فيرويس

Top 50 Products By Total Number Of "Distinct" Vulnerabilities in 2017

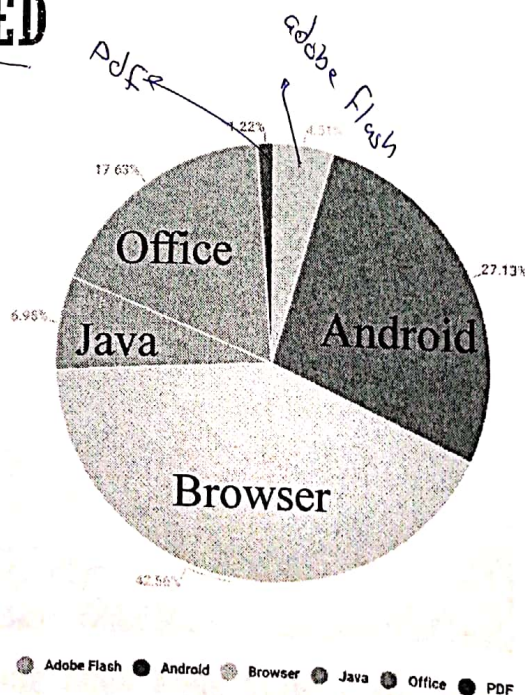
Go to year: 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008 2009 2010 2011 2012 2013

Time Leaders

	Product Name	Vendor Name	Product Type	Number of Vulnerabilities
1	Android	Google	OS	842
2	Linux Kernel	Linux	OS	453
3	Iphone Os	Apple	OS	387
4	Imagemagick	Imagemagick	Application	357
5	Mac Os X	Apple	OS	299
6	Windows 10	Microsoft	OS	268
7	Windows Server 2016	Microsoft	OS	252
8	Windows Server 2008	Microsoft	OS	243
9	Windows Server 2012	Microsoft	OS	235
10	Debian Linux	Debian	OS	230
11	Windows 7	Microsoft	OS	229
12	Windows 8.1	Microsoft	OS	225

source: <https://www.cvedetails.com/top-50-products.php?year=2017>

VULNERABLE APPLICATIONS BEING EXPLOITED



Source: Kaspersky Security Bulletin 2017

* PHILOSOPHY OF THIS COURSE

- We are not going to be able to cover everything

- We are not going to be able to cover everything

- Main goal: You will *not* be a security expert after this class

- A sample (after this class, you should realize why it

- The security would be dangerous to think you are)

- Become familiar with basic acronyms (ASAC, BOB, FCI, etc.) and "buzzwords" (phishing)

- Become

- Try to have a better appreciation of security issues after this class

Course Organization

A NAÏVE VIEW

IP Hack (Manipulate)



objectives of Security

read data

Computer security is about **CIA!**

Confidentiality, integrity, and availability → IP Hacked (Denial of Service)

use (Encryption) use (Hash Func)

These are important, but security is about much more...

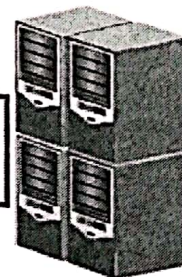
Cryptographic (Key)

The legitimate users ~~are~~
 The users can get the services when they need it.

A NAÏVE VIEW



password



Requirements

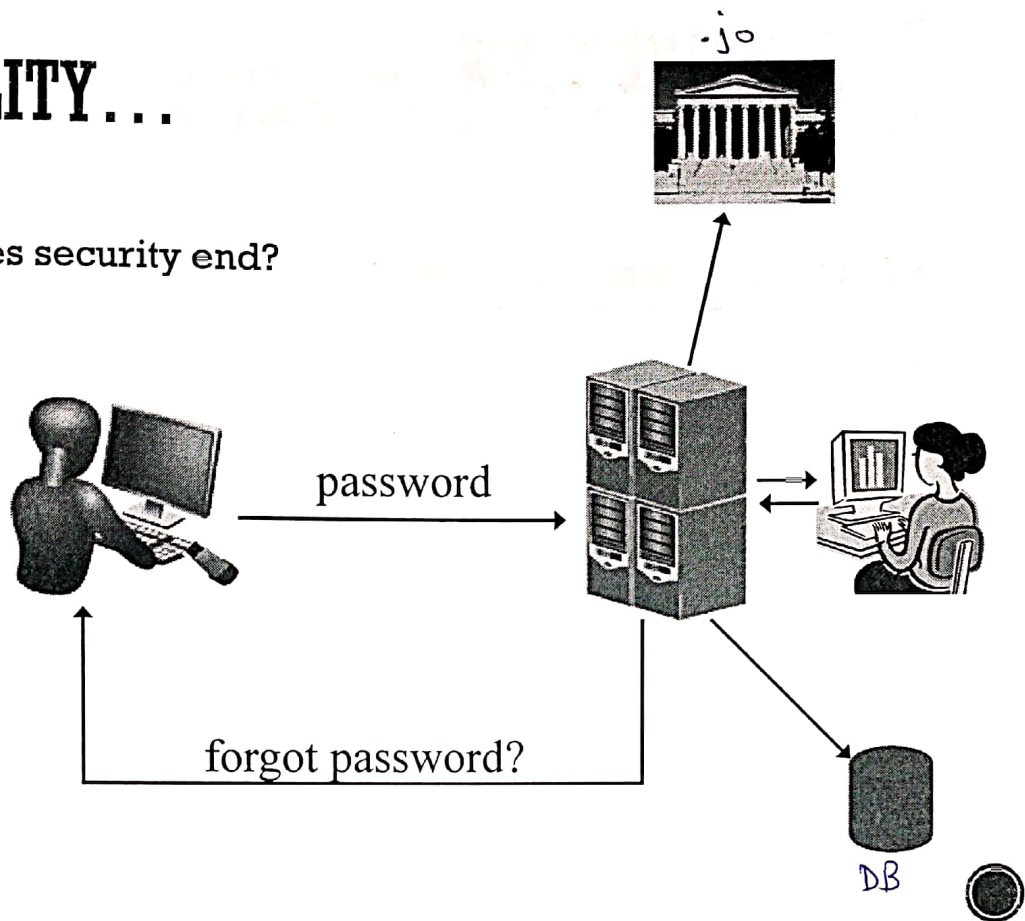
Functional

non-functional

Security

IN REALITY...

- Where does security end?



SSN: Social security number (SSN)

ONE GOOD ATTACK

- Use public records to figure out someone's password
 - Or, e.g., their SSN, so can answer security question...
- The problem is *not* (necessarily) that SSNs are public
- The problem *is* that we "overload" SSNs, and use them for more than they were intended

A NAÏVE VIEW

- Achieve "absolute" security

IN REALITY...

- Absolute security is easy to achieve!
 - How...?
- Absolute security is impossible to achieve!
 - Why...?
- Good security is about risk management
 - ← How?
 - types of risks
 - possibility of each risk
 - Cost of ~~the~~ risk if happened
 - Cost of preventing risk.

SECURITY AS A TRADE-OFF

- The goal is not (usually) "to make the system as secure as possible"...
- ...but instead, "to make the system as secure as possible within certain constraints" (cost, usability, convenience)
- Must understand the existing constraints → سهولة
ملائمة
 - E.g., passwords...

COST-BENEFIT ANALYSIS

- Important to evaluate what level of security is necessary/appropriate
 - Cost of mounting a particular attack vs. value of attack to an adversary
 - Cost of damages from an attack vs. cost of defending against the attack
 - Likelihood of a particular attack
- Sometimes the best security is to make sure you are not the easiest target for an attacker...

"MORE" SECURITY NOT ALWAYS BETTER

- "No point in putting a higher post in the ground when the enemy can go around it"
- Need to identify the weakest link
 - Security of a system is only as good as the security at its weakest point...
- Security is not a "magic bullet"
- Security is a process, not a product

* COMPUTER SECURITY IS NOT JUST ABOUT SECURITY

- Detection, response, audit
 - How do you know when you are being attacked?
 - How quickly can you stop the attack?
 - ↳ Can you identify the attacker(s)?
 - ↳ Can you prevent the attack from recurring?
- Recovery
 - Can be much more important than prevention
- Economics, insurance, risk management..
- Offensive techniques

COMPUTER SECURITY IS NOT JUST ABOUT COMPUTERS

- What is "the system"?
- Physical security → حماية خارجية للاجهزة
- Social engineering → use non-technical ways to attack
 - Bribes for passwords رشوة
 - Phishing ← طريقة كاذب → like, attacker sends you an email pretending to be your bank for example and asks you to follow some link and steps
- "External" means of getting information
- Legal records
- Trash cans

SECURITY MINDSET

- Learn to think with a "security mindset" in general
 - What is "the system"?
 - How could this system be attacked?
 - What is the weakest point of attack?
 - How could this system be defended?
 - What threats am I trying to address?
 - How effective will a given countermeasure be?
 - What is the trade-off between security, cost, and usability?

SUMMARY

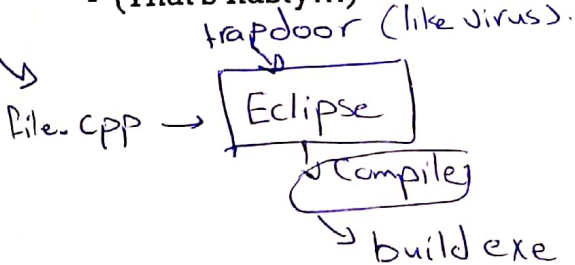
- "The system" is not just a computer or a network
- Prevention is not the only goal
 - Cost-benefit analysis
 - Detection, response, recovery
- Nevertheless...in this course, we will focus on *computer security*, and primarily on *prevention*
 - If you want to be a security expert, you need to keep the rest in mind

COMPUTERS ARE EVERYWHERE...

- ...and can always be attacked
- Electronic banking, social networks, e-voting
- iPods, iPhones, PDAs, RFID transponders
- Automobiles
- Appliances, TVs
- (Implantable) medical devices
- Cameras, picture frames(!)
 - See <http://www.securityfocus.com/news/11499>

"TRUSTING TRUST"

- Consider a compiler that embeds a trapdoor into anything it compiles
- How to catch?
 - Read source code? (What if replaced?)
 - Re-compile compiler?
- What if the compiler embeds the trojan code whenever it compiles a compiler?
 - (That's nasty...)



منه حصان طروادة
(ملف جتروا على سطر)
لانه جواره كود

"TRUSTING TRUST"

- Whom do you trust?
- Does one really need to be this paranoid??
 - Probably not
 - Sometimes, yes
- Shows that security is complex...and essentially impossible
- Comes back to risk/benefit trade-off

Next time:
begin cryptography

COMPUTER AND NETWORK SECURITY

LECTURE 2

Jonathan Katz

Modified By: Dr. Ramzi Saifan



A high-level survey of cryptography

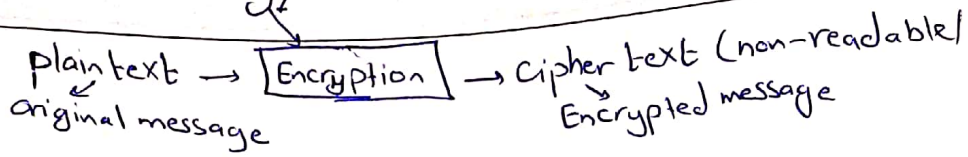
GOALS OF CRYPTOGRAPHY

- Crypto deals primarily with three goals:

- Confidentiality ^{سرية}
- Integrity (of data) ^{سلامة البيانات}
- Authentication (of resources, people, systems) ^{معرفة من هو الشخص صاحب البيانات}

- Other goals also considered

- E.g., non-repudiation → If someone sends a message he can't deny it.
- Accountability → someone is responsible for action.
- Anonymity → you can't know me or where I am from any action or message
- ...



CRYPTOGRAPHIC SYSTEMS

Characterized along three independent dimensions:

Categories of cryptographic system

① types of operation

② The type of operations used for transforming plaintext to ciphertext

The number of keys used

③ The way in which the plaintext is processed

original	crypt-
T	a
b	c

بید کی حرف بجز
تایف باج

Substitution

Symmetric, single-key, secret-key, conventional encryption

Block cipher

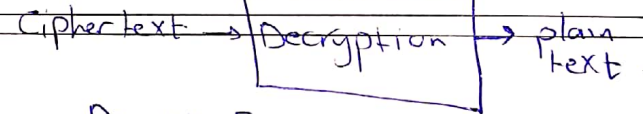
Transposition

Asymmetric, two-key, or public-key encryption

Stream cipher

بید اماکن لاجز
عنا اخی الج غیر
مقوده بنفس
المسج اللیل

both low complexity & fast
reversible



Enc \rightarrow k_1 استمر $k_2 \neq k_1$ فكس -- Decry \rightarrow Enc \rightarrow key k_1 فكس
 Dec $\rightarrow k_2$

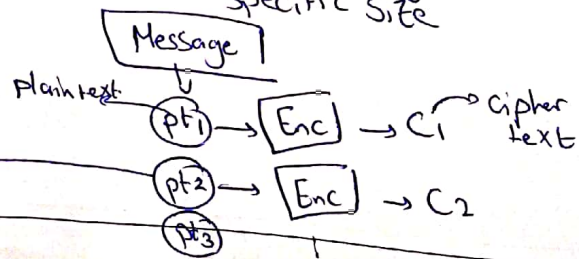
* PRIVATE- VS. PUBLIC-KEY SETTINGS

- For the basic goals, there are two settings:
 - Private-key / shared-key / symmetric-key / secret-key
 - Public-key
- The private-key setting is the "classical" one (thousands of years old)
- The public-key setting dates to the 1970s

* stream cipher message Not divided.

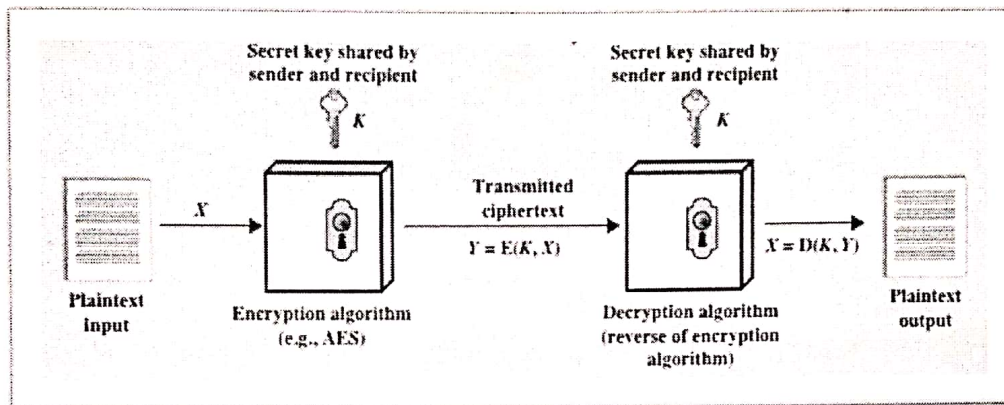
* Block Cipher: Divide message into blocks of specific size

Block size is according to the Encryption Algorithm



PRIVATE-KEY CRYPTOGRAPHY

- The communicating parties share some information that is random and secret
 - This shared information is called a key
 - ↳ Key is not known to an attacker
 - This key must be shared (somehow) in advance of their communication



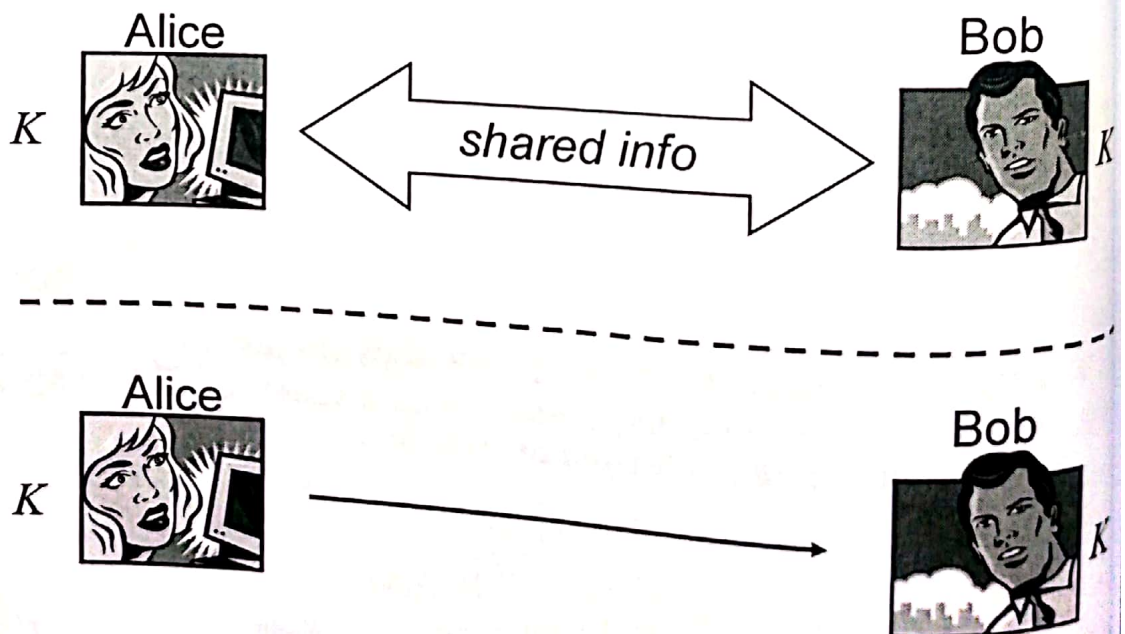
TO EMPHASIZE

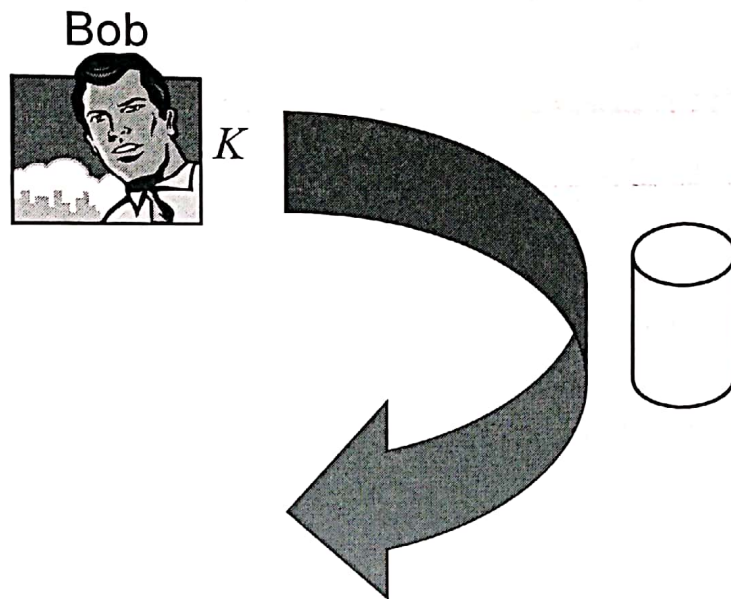
- Alice and Bob share a key K
 - Must be shared securely
 - Must be completely random
 - Must be kept completely secret from attacker
- We don't discuss (for now) how they do this
 - ↳ You can imagine they meet on a dark street corner and Alice hands a USB device (with a key on it) to Bob

CANONICAL APPLICATIONS

عز آمن

- Two (or more) distinct parties communicating over an insecure network
 - E.g., secure communication
- A single party who is communicating "with itself" over time
 - E.g., secure storage





SECURITY THROUGH OBSCURITY?

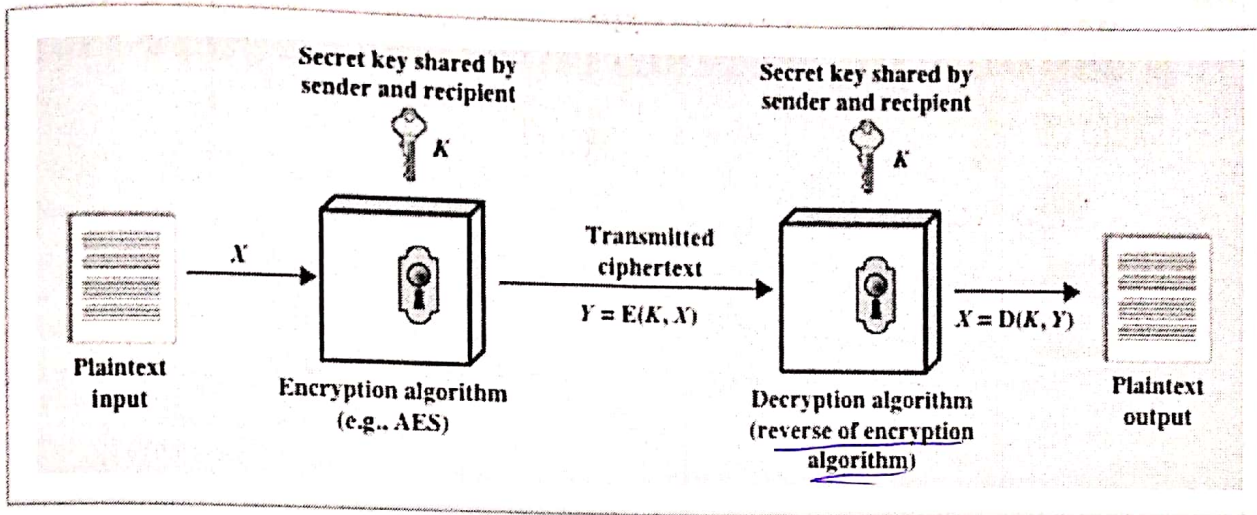
- Always assume that the full details of crypto protocols and algorithms are public
 - Known as **Kerckhoffs' principle**
 - The only secret information is a key
- "Security through obscurity" is a bad idea...
 - True in general; even more true in the case of cryptography
 - Home-brewed solutions are BAD!
 - Standardized, widely-accepted solutions are GOOD!

SECURITY THROUGH OBSCURITY?

- Why not?
- Easier to maintain secrecy of a *key* than an *algorithm*
 - Reverse engineering
 - Insider attacks
- Easier to change the key than the algorithm
- In general setting, much easier to share an algorithm than for everyone to use their own

Private-key encryption

Functional definition



Encryption: $c \leftarrow E_K(m)$ possibly randomized!
Handwritten notes: 'Cypher' points to 'c', 'message' points to 'm', 'Encryption' points to 'E', 'key' points to 'K'.

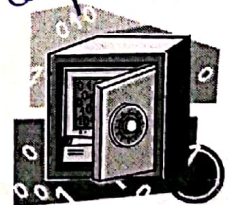
Decryption algorithm: $m = D_K(c)$
Handwritten note: 'Decryption' points to 'D'.

Correctness: for all K, we have $D_K(E_K(m)) = m$ ~~reversible~~ reversible

ENCRYPTION SCHEME SECURITY

- Unconditionally secure \rightarrow you can't break it.
 - No matter how much time an opponent has, it is impossible for him or her to decrypt the ciphertext simply because the required information is not there
- Computationally secure
 - The cost of breaking the cipher exceeds the value of the encrypted information
 - The time required to break the cipher exceeds the useful lifetime of the information

* you can break it if you have enough resources $\left\{ \begin{array}{l} \text{money} \\ \text{time} \\ \text{Computation} \end{array} \right.$
 you need resources more than the cost of the information



MODEL OF SYMMETRIC CRYPTOSYSTEM

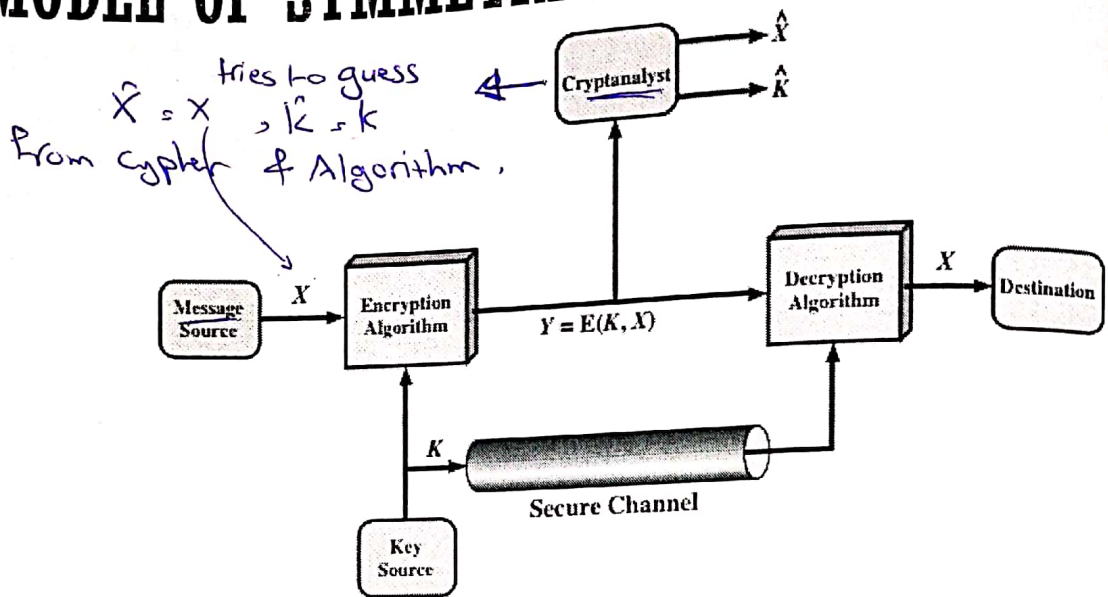


Figure 2.2 Model of Symmetric Cryptosystem

CRYPTANALYSIS AND BRUTE-FORCE ATTACK

two ways to break Encryption Algorithm

① Cryptanalysis

- Attack relies on the nature of the algorithm plus some knowledge of the general characteristics of the plaintext
- to attempt to deduce a specific plaintext or to deduce the key being used

* don't use simple Algorithms

② Brute-force attack

- Attacker tries every possible key on a piece of ciphertext until an intelligible translation into plaintext is obtained
- On average, half of all possible keys must be tried to achieve success.
- To supplement the brute-force approach, some degree of knowledge about the expected plaintext is needed,

* Increase key size

Type of Attack	Known to Cryptanalyst
Ciphertext Only	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext
Known Plaintext	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext • One or more plaintext-ciphertext pairs formed with the secret key
Chosen Plaintext	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext • Plaintext message chosen by cryptanalyst, together with its corresponding ciphertext generated with the secret key
Chosen Ciphertext	<ul style="list-style-type: none"> • Encryption algorithm • Ciphertext • Ciphertext chosen by cryptanalyst, together with its corresponding decrypted plaintext generated with the secret key
Chosen Text	<ul style="list-style-type: none"> • 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

TABLE 2.1
TYPES OF
ATTACKS
ON
ENCRYPTED
MESSAGES

A CLASSIC EXAMPLE: SHIFT CIPHER

Encryption : $C = (M + k) \bmod{n}$ number of letters
 Decryption : $M = (C - k) \bmod{n}$

- Assume the English uppercase alphabet (no lowercase, punctuation, etc.)
 - View letters as numbers in $\{0, \dots, 25\}$
- The key is a random letter of the alphabet
- Encryption done by addition modulo 26
- Is this secure?
 - Exhaustive key search
 - Automated determination of the key

A B C D E F
 C D E F A B
 shift

BRUTE-FORCE CRYPTANALYSIS OF SHIFT CIPHER

text found
(key) →

KEY	PHHW	PH	DIWHU	WKH	WRJD	SDIWB
1	oggv	og	chvgt	vjg	vgic	rectva
2	nffu	nf	bgufs	uif	uphb	qbsuz
3	meet	me	after	the	toga	party
4	ldds	ld	zesdq	sgd	snfz	ozgsx
5	kccr	kc	ydrp	rfe	rmey	nyprw
6	jbbq	jb	xcqbo	qeb	qldx	mxoqv
7	iaap	ia	wbpan	pda	pkcw	lwnpu
8	hzzo	hz	vaozm	ocz	objv	kvmot
9	gyyn	gy	uznyl	nby	niau	julns
10	fxxm	fx	tymxk	max	mhzt	itkmr
11	ewwl	ew	sxlwj	lzw	lgys	hsjlg
12	dvvk	dv	rkwvi	kyv	kfxr	grikp
13	cuuj	cu	qvjuh	jxu	jewq	fqhjo
14	btti	bt	puitg	iwt	idvp	epgin
15	assh	as	othsf	hvs	hcuo	dofhm
16	zrrg	zr	nsgre	gur	gbtn	cnegl
17	yqaf	yq	mrfqd	ftq	fasm	bmdfk
18	xppe	xp	lqepc	esp	ezrl	alcej
19	wood	wo	kpdob	dro	dyqk	zkbdi
20	vnnc	vn	jocna	cqn	cxpj	yjach
21	ummb	um	inbmz	bpm	bwoi	xizbg
22	tlla	tl	hmaly	aol	avnh	whyaf
23	skkz	sk	glzxx	zkn	zumg	vgxze
24	rjyy	rj	fkyjw	ymj	ytlf	ufwyd
25	qiix	qi	ejxiv	xli	xske	tevxc

Figure 2.3 Brute-Force Cryptanalysis of Caesar

MONOALPHABETIC CIPHER

- The key is a random permutation of the alphabet
 - Note: key space is huge!
- Encryption done in the natural way

▪ Is this secure?

- Frequency analysis

▪ A large key space is necessary, but not sufficient, for security

(e.g) using Mono-alphabetic cipher of 26 English letters, how long do we need to break it using brute-force attack given a submachine gun tool that can try 1000 key/sec

Sol:

$$\frac{26!}{1000} \times 24 \text{ hours} \rightarrow \text{days}$$

356 × 1000 × 3600 × 24

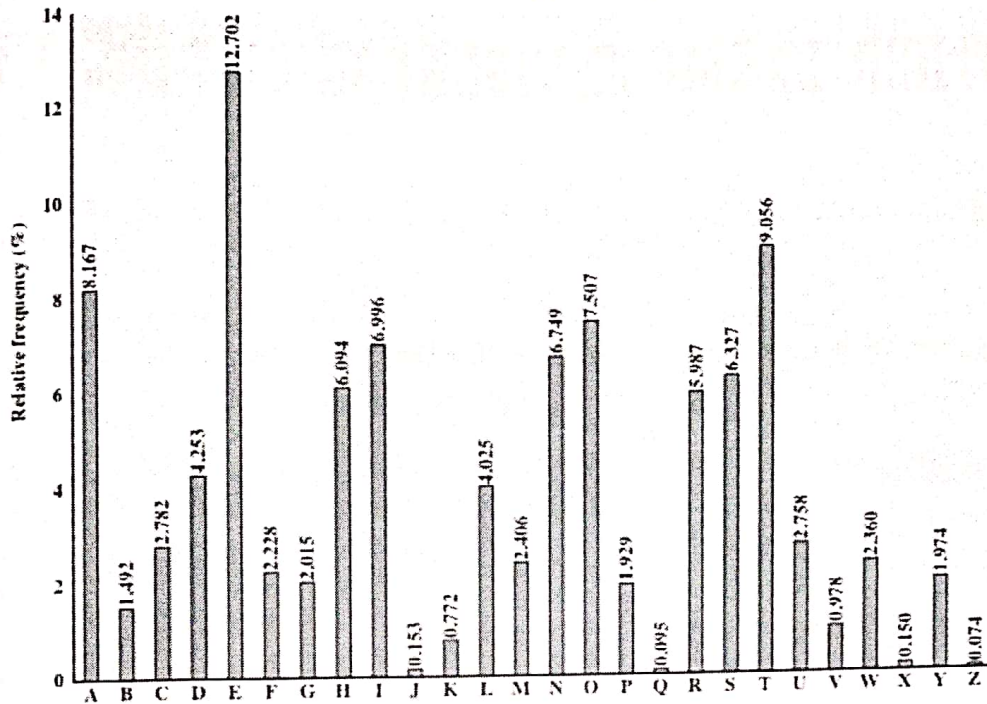
years

If you want hours

→ If I know cypher text & plain text → I can know the key.

For the past example maybe I can know the key depending on frequency of letters

مع الاتی بار text حروف متكررة اكثر من غيرها .



Week 3 lectures

Figure 2.5 Relative Frequency of Letters in English Text

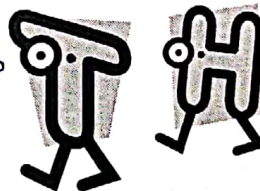
MONOALPHABETIC CIPHERS

- Easy to break because they reflect the frequency data of the original alphabet
- Countermeasure is to provide multiple substitutes (homophones) for a single letter

Digram يعني استبدال كل حرفين بحرفين

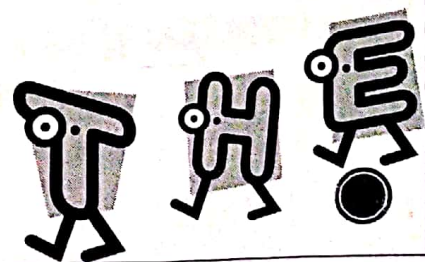
- Two-letter combination
- Most common is *th*

Key size 26×26



Trigram

- Three-letter combination
- Most frequent is *the*



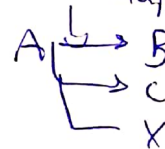
ANOTHER EXAMPLE: VIGENERE CIPHER

- More complicated version of shift cipher
- Believed to be secure for over 100 years
- Is it secure?

POLYALPHABETIC CIPHERS

كل مرة يستبدل حرف مختلف (متعدد)

We need method to choose mapped letter



- Polyalphabetic substitution cipher
 - Improves on the simple monoalphabetic technique by using different monoalphabetic substitutions as one proceeds through the plaintext message

All these techniques have the following features in common:

- A set of related monoalphabetic substitution rules is used
- A key determines which particular rule is chosen for a given transformation

VIGENÈRE CIPHER

- Best known and one of the simplest polyalphabetic substitution ciphers
- In this scheme the set of related monoalphabetic substitution rules consists of the 26 Caesar ciphers with shifts of 0 through 25
- Each cipher is denoted by a key letter which is the ciphertext letter that substitutes for the plaintext letter a

EXAMPLE OF VIGENÈRE CIPHER

- To encrypt a message, a key is needed that is as long as the message
- Usually, the key is a repeating keyword
- For example, if the keyword is deceptive, the message "we are discovered save yourself" is encrypted as:

key:

deceptivedeceptivedeceptive

plaintext: **wearediscoveredsaveyourself**

ciphertext: **ZICVTWQNGRZGVTWAVZHCQYGLMGJ**

VIGENERE AUTOKEY SYSTEM

- A keyword is concatenated with the plaintext itself to provide a running key
- Example:
 key: deceptivewere discovered save
 plaintext: we are discovered save yourself
 ciphertext: ZICVTWQNGKZEIIGASXSTSLVWVLA
- Even this scheme is vulnerable to cryptanalysis
 - Because the key and the plaintext share the same frequency distribution of letters, a statistical technique can be applied

*Vigener cipher is not ~~safe~~ secure against frequency analysis - but we have to do pre-processing to know the length of the key

- 1) know the length of the keyword
- 2) do frequency analysis

2 ways :-
 + We are discovered save yourself
 + keykeykeykeykeykeykeykeykeykey
 %26

ATTACKING THE VIGENERE CIPHER

positions
 0, 3, 6, 9, 12, 15, ... → Enc using K
 1, 4, 7, 10, 13, 16, 19 → Enc using E

- Let p_i (for $i=0, \dots, 25$) denote the frequency of letter i in English-language text
 - Known that $\sum p_i^2 \approx 0.065$
- For each candidate period t , compute frequencies $\{q_i\}$ of letters in the sequence c_0, c_t, c_{2t}, \dots
- For the correct value of t , we expect $\sum q_i^2 \approx 0.065$
 - For plain text, MonoAlpha Cipher
- For incorrect values of t , we expect $\sum q_i^2 \approx 1/26$
 - For random text
- Once we have the period, can use frequency analysis as in the case of the shift cipher

freq analysis

→ 2, 5, 8, 11, 14, 17, 20, ... → Enc using Y
 Cipher text if frequency is not maintained ⇒

$$\sum_{i=0}^{25} p_i^2 = 1/26$$

MORAL OF THE STORY?

- Don't use "simple" schemes
- Don't use schemes that you design yourself
 - Use schemes that other people have already designed and analyzed...

* assume Key length = 1, If shift cipher, find C_i for each letter in cipher text.

$$\text{If } \sum_{i=0}^{25} (C_i)^2 \approx 0.065 \text{ then Key length} = 1 \\ \text{else Key length} \neq 1$$

* assume Key length = 2
0, 2, 4, 6, 8, ... Freq C_{i1}
1, 3, 5, 7, 9, ... Freq C_{i2}

$$\sum_{i=0}^{25} \frac{(C_{i1})^2}{(C_{i2})^2} \approx 0.065$$

} assume many Key length until you find the right one

A FUNDAMENTAL PROBLEM

- A fundamental problem with "classical" cryptography is that *no definition of security was ever specified*
 - It was not even clear what it meant for a scheme to be "secure"
- As a consequence, *proving security was not even an option*
 - So how can you *know* when something is secure?
 - (Or is at least based on well-studied, widely-believed assumptions)

SECURITY GOALS?

- Adversary unable to recover the key
 - Necessary, but meaningless on its own...
- Adversary unable to recover entire plaintext
 - Good, but is it enough?
- Adversary unable to determine any information at all about the plaintext
 - Formalize?
 - Sounds great!
 - Can we achieve it?

CMSC 414 Computer and Network Security

Lecture 3

Jonathan Katz
Modified by: Dr. Ramzi Saifan

Perfect secrecy

unconditional secure

Defining secrecy (take 1)

- Even an adversary running for an unbounded amount of time learns nothing about the message from the ciphertext
 - (Except the length)
- *Perfect secrecy*
- Formally, for all distributions over the message space, all m , and all c :

$$\Pr[M=m \mid C=c] = \Pr[M=m]$$

probability

message equals
specific message

the attacker
knows the
cipher text.

means Cipher text gives No
information at all
about the plain text.

Properties of the one-time pad?

- Achieves perfect secrecy
 - No eavesdropper (no matter how powerful) can determine any information whatsoever about the plaintext
- was developed by Gilbert Vernam in 1918.
- Stream cipher: The message is represented as a binary string.
- The key is a truly random sequence of 0's and 1's of the same length as the message.
- The encryption is done by XOR the key and the message.

→ to be truly random } → must be unpredictable
 } → balanced

given the bits b_0, b_1, \dots, b_i

$$P(b_i = 1) = \frac{1}{2} \rightarrow P(b_i = 1) = \frac{1}{2}$$

$$C_i = m_i \oplus k_i$$

For Decryption

$$m_i = c_i \oplus k_i$$

Why OTP is perfect secure?

- The security depends on the randomness of the key.
- In cryptographic context, we seek two fundamental properties in a binary random key sequence:
 - Unpredictability: the probability of a certain bit being 1 or 0 is exactly equal to $\frac{1}{2}$ even if you have all previous bits.
 - Balanced (Equal Distribution):
 - The number of 1's and 0's should be equal.

2-3

Mathematical Proof

- the probability of a key bit being 1 or 0 is exactly equal to $\frac{1}{2}$.
- The plaintext bits are not balanced. Let the probability of 0 be x and then the probability of 1 turns out to be $1-x$.
- Let us calculate the probability of ciphertext bits.

Mathematical Proof

Week 4 lectures.

m_i prob.	k_i prob.	c_i prob.
0 x	0 $\frac{1}{2}$	0 $\frac{1}{2}x$
0 x	1 $\frac{1}{2}$	1 $\frac{1}{2}x$
1 $1-x$	0 $\frac{1}{2}$	1 $\frac{1}{2}(1-x)$
1 $1-x$	1 $\frac{1}{2}$	0 $\frac{1}{2}(1-x)$

- We find out the probability of a ciphertext bit being 1 or 0 is equal to $(\frac{1}{2})x + (\frac{1}{2})(1-x) = \frac{1}{2}$.
Ciphertext looks like a random sequence.

The ciphertext shape and content doesn't depend on the message

Disadvantages

- (Essentially) useless in practice...
 - Long key length
 - Can only be used once (hence the name!)
 - Insecure against known-plaintext attacks
 - Key distribution & Management difficult.
- These are inherent limitations of perfect secrecy

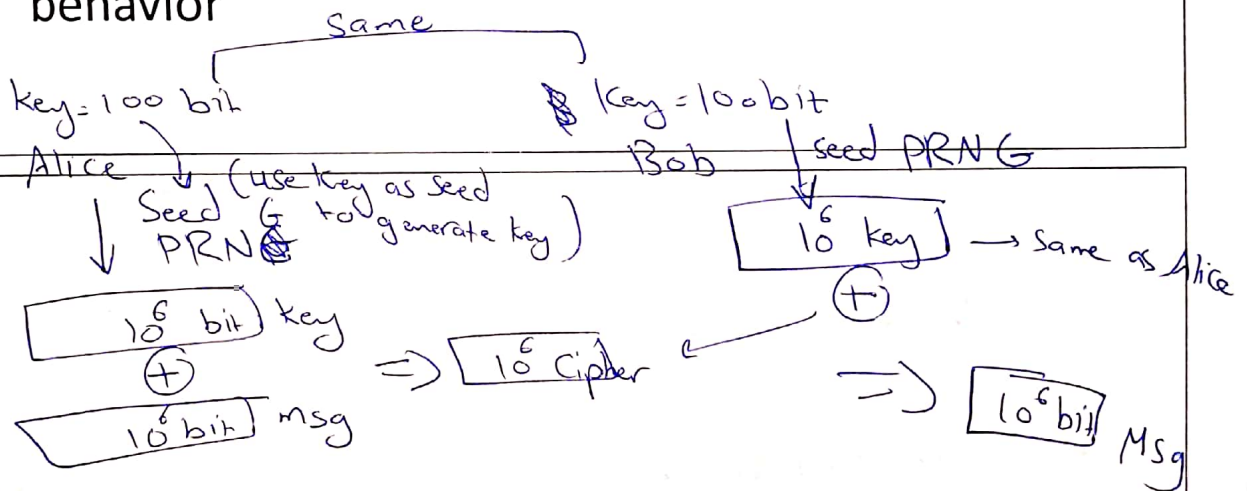
A computationally secure scheme

- A pseudorandom (number) generator (PRNG) is a deterministic function that takes as input a seed and outputs a string
- If seed chosen at random, output of the PRNG should "look random" (i.e., be pseudorandom)

To use one time pad, use the key as a seed to pseudo Random Number Generator
⇒ generate key of length = msg length.

Notes

- Pseudo-randomness must be indistinguishable from random for all efficient algorithms
 - General-purpose PRNGs *not sufficient* for crypto
- Pseudorandomness of the PRNG depends on the seed being chosen “at random”
 - Note in particular that if a seed is re-used then the output of the PRNG remains the same!
 - In practice: from physical processes and/or user behavior



Computational secrecy

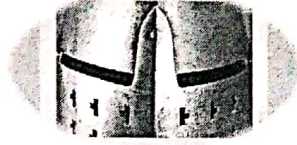
Computational secrecy

- We can overcome the limitations of perfect secrecy by (slightly) relaxing the definition
- Instead of requiring *total* secrecy against *unbounded* adversaries, require secrecy against *time-bounded* adversaries except with some *small probability*
 - E.g., ^{Secure} secrecy for 100 years, except with probability
 2^{-80} → هكره
Hack it

* The take-home message

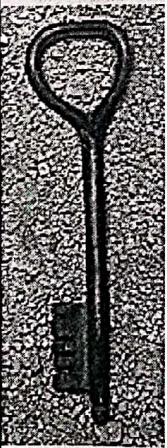
- Weakening the definition slightly allows us to construct much more efficient schemes!
- Strictly speaking, no longer 100% absolutely guaranteed to be secure
 - Security of encryption now depends on security of building blocks (which are analyzed extensively, and are believed to be secure)
 - Given enough time and/or resources, the scheme can be broken

* Revise Stream cipher First!



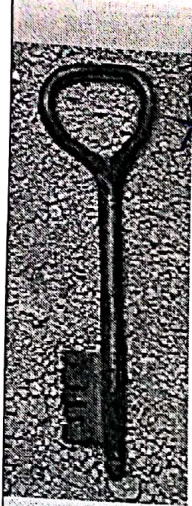
Block Ciphers and the Data Encryption Standard (DES)

Modified by: Dr. Ramzi Saifan



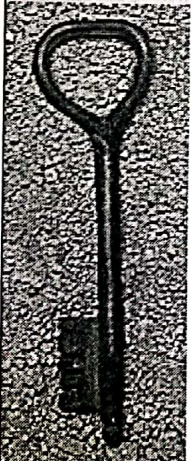
Block ciphers

- ◆ Keyed, invertible
- ◆ Large key space, large block size → determined by Encryp- algorithm.
- ◆ A block of plaintext is treated as a whole and used to produce a ciphertext block of equal length
- ◆ Typically a block size of 64 or 128 bits is used
- ◆ The majority of network-based symmetric cryptographic applications make use of block ciphers



* Data Encryption Standard (DES)

- ◆ Developed in 1970s by IBM / NSA / NBS
 - Non-public design process
- ◆ Block size = 64-bit input/output *Cipher text size = 64 bits*
- ◆ Key size = 56 bits out of a 64 bits *Key space = 2⁵⁶*
 - One bit in each octet is a parity-check bit
- ◆ Was the most widely used encryption scheme until the introduction of the Advanced Encryption Standard (AES) in 2001



Feistel Cipher

- ◆ Proposed the use of a cipher that alternates substitutions and permutations

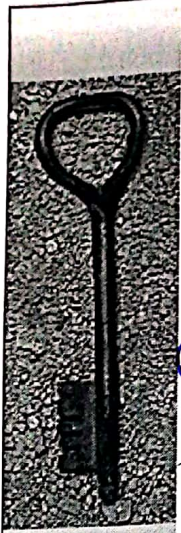
* Substitutions

- Each plaintext element or group of elements is uniquely replaced by a corresponding ciphertext element or group of elements

* Permutation

- No elements are added or deleted or replaced in the sequence, rather the order in which the elements appear in the sequence is changed

Is the structure used by many significant symmetric block ciphers currently in use.



Feistel Cipher Structure

Using a key we generate Round Keys

$$LE_i = RE_{i-1}$$

$$RE_i = LE_{i-1} \oplus F(K_i, RE_{i-1})$$

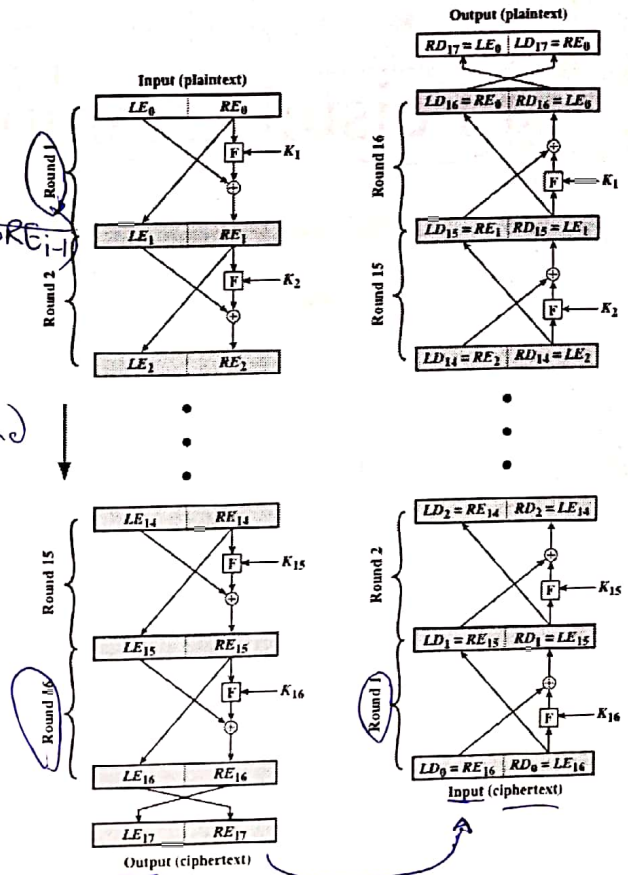
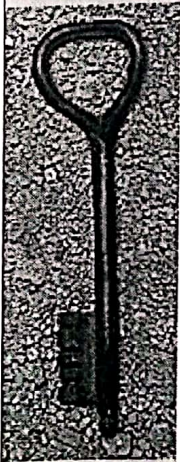
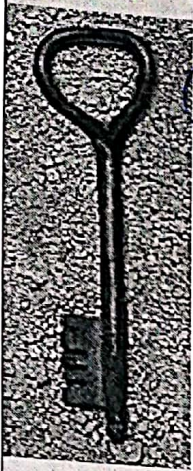


Figure 3.3 Feistel Encryption and Decryption (16 rounds)



Feistel Cipher Design Features

- ◆ Block size
 - Larger block sizes mean greater security but reduced encryption/decryption speed for a given algorithm
- ◆ Key size
 - Larger key size means greater security but may decrease encryption/decryption speeds
- ◆ Number of rounds
 - The essence of the Feistel cipher is that a single round offers inadequate security but that multiple rounds offer increasing security
- ◆ Subkey generation algorithm
 - Greater complexity in this algorithm should lead to greater difficulty of cryptanalysis
- ◆ Round function F
 - Greater complexity generally means greater resistance to cryptanalysis
- ◆ Fast software encryption/decryption
 - In many cases, encrypting is embedded in applications or utility functions in such a way as to preclude a hardware implementation; accordingly, the speed of execution of the algorithm becomes a concern
- ◆ Ease of analysis
 - If the algorithm can be concisely and clearly explained, it is easier to analyze that algorithm for cryptanalytic vulnerabilities and therefore develop a higher level of assurance as to its strength



Feistel Example

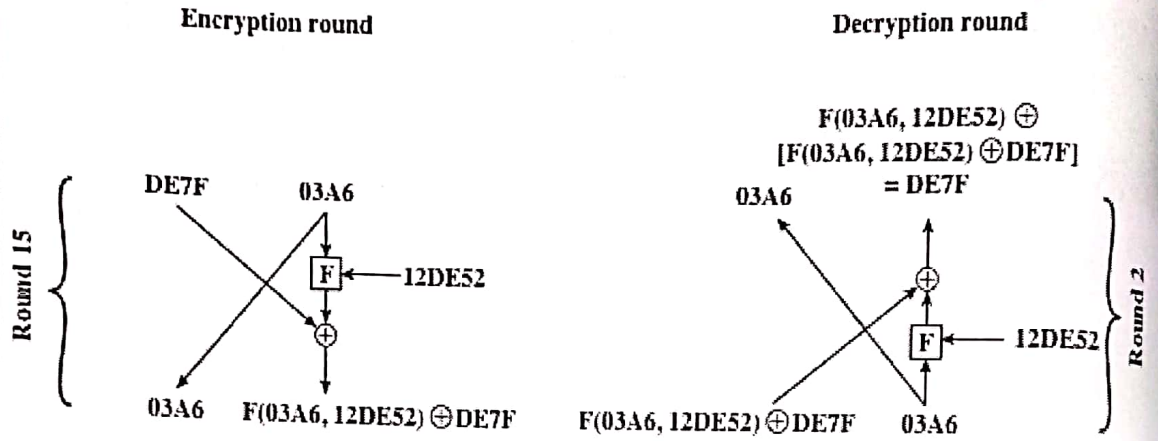
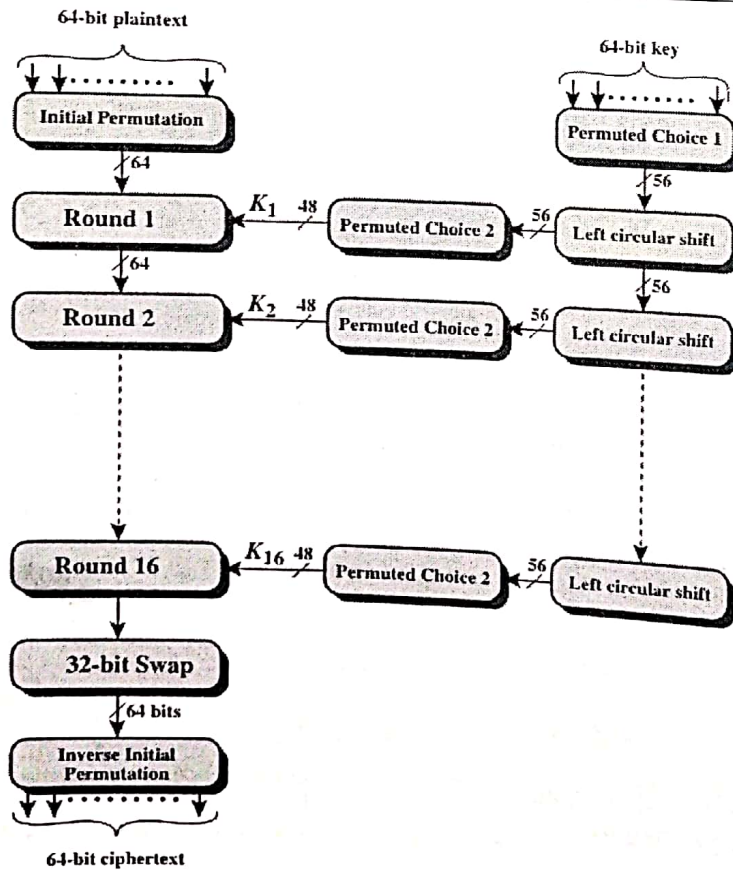
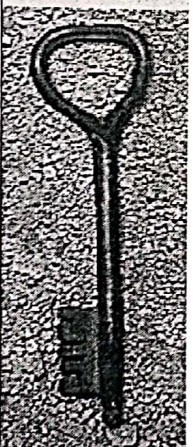


Figure 3.4 Feistel Example



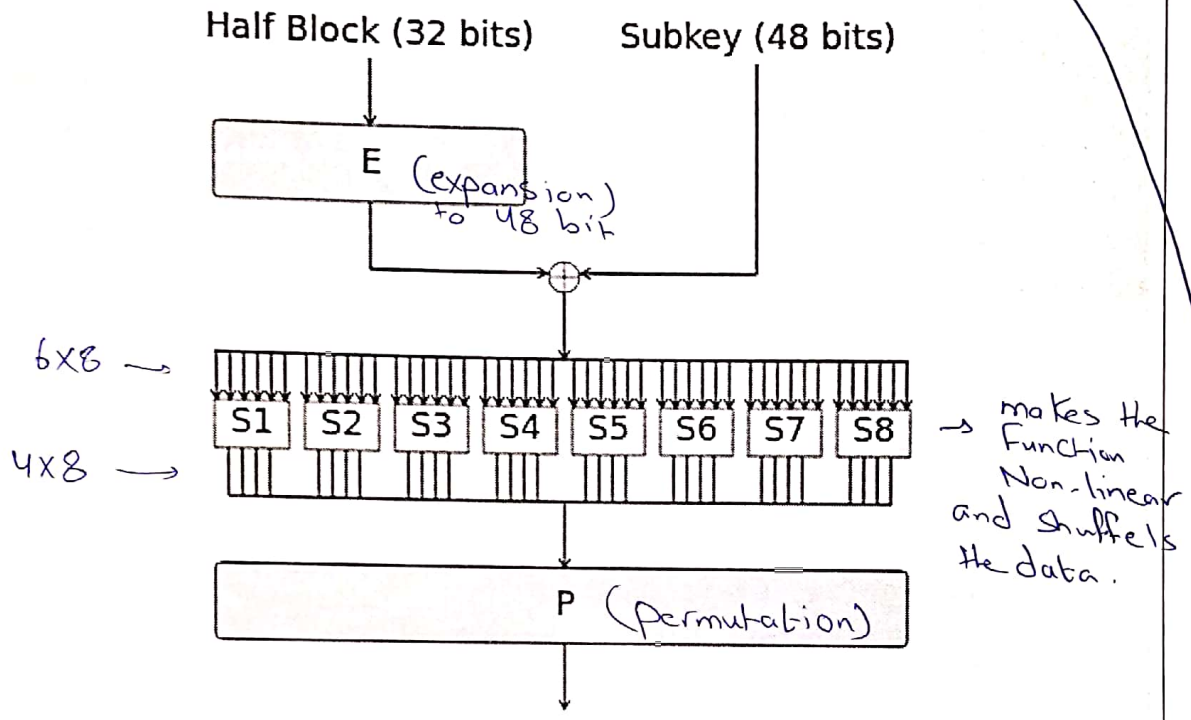
DES Encryption Algorithm

Figure 3.5 General Depiction of DES Encryption Algorithm

DES:

bruteforce attack : 2^{56} possible keys
 If Computer can do 10^9 dec/sec? $\frac{2^{56}}{10^9/2}$ on average

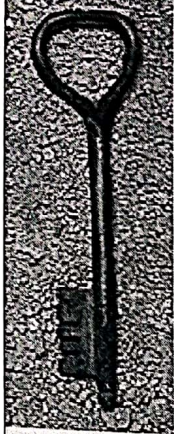
Round Function



Average Time Required for Exhaustive Key Search

Key size (bits)	Cipher	Number of Alternative Keys	Time Required at 10^9 decryptions/s	Time Required at 10^{13} decryptions/s
56	DES	$2^{56} \approx 7.2 \times 10^{16}$	2^{55} ns = 1.125 years	1 hour
128	AES	$2^{128} \approx 3.4 \times 10^{38}$	2^{127} ns = 5.3×10^{21} years	5.3×10^{17} years
168	Triple DES	$2^{168} \approx 3.7 \times 10^{50}$	2^{167} ns = 5.8×10^{33} years	5.8×10^{29} years
192	AES	$2^{192} \approx 6.3 \times 10^{57}$	2^{191} ns = 9.8×10^{40} years	9.8×10^{36} years
256	AES	$2^{256} \approx 1.2 \times 10^{77}$	2^{255} ns = 1.8×10^{60} years	1.8×10^{56} years
26 characters (permutation)	Monoalphabetic	$26! \approx 4 \times 10^{26}$	2×10^{26} ns = 6.3×10^9 years	6.3×10^6 years

Block Cipher Design Principles: Design of Function F



- ◆ The heart of a Feistel block cipher is the function F
- ◆ The more nonlinear F, the more difficult any type of cryptanalysis will be
- ◆ The SAC and BIC criteria appear to strengthen the effectiveness of the confusion function

The algorithm should have good avalanche properties

Strict avalanche criterion (SAC)

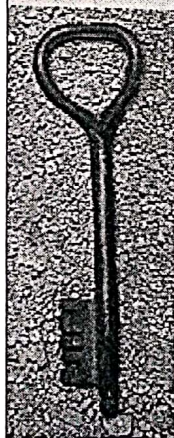
States that any output bit j should change with probability $1/2$ when any single input bit i is inverted for all i, j

Bit independence criterion (BIC)

States that output bits j and k should change independently when any single input bit i is inverted for all i, j , and k

↳ Avalanche Criteria
change of any bit in the inputs
will change many bits in the output

Concerns about DES

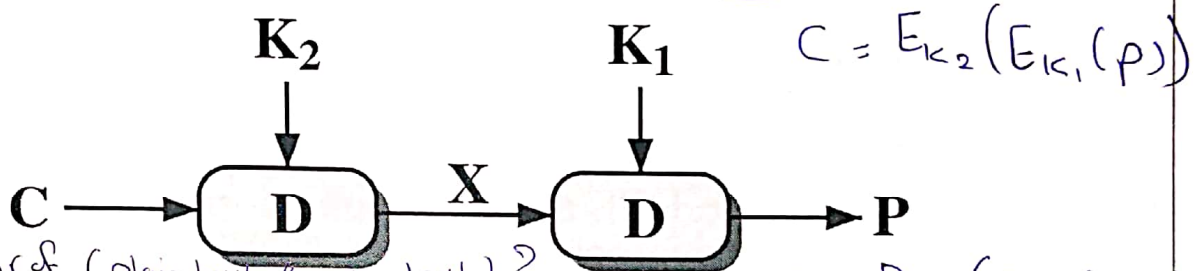
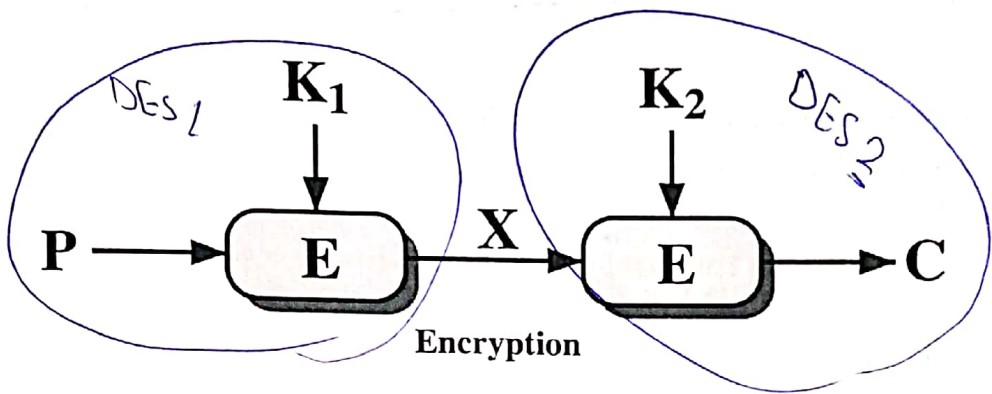


- ◆ Short key length
 - DES “cracker”, can break DES in days
 - Computation can be distributed to make it faster
 - Does not mean “DES is insecure”; depends on desired security
- ◆ Short block length
 - Repeated blocks happen “too frequently”
- ◆ Some (theoretical) attacks have been found
 - Claimed known to DES designers 15 years before public discovery!

Key = $k_1 || k_2$
 $56 || 56 = 112$ bits
 Can concatenated

$\frac{2^{112}}{2}$ on average trials to break the key

Double DES



$$C = E_{K_2}(E_{K_1}(P))$$

$$P = D_{K_2}(D_{K_1}(C))$$

Given pair of (plain text & cipher text) →
 ① Decrypt the cipher text using all possible keys (2^{56} trials)

② Encrypt M using all possible key (2^{56} trials) (a) Double Encryption

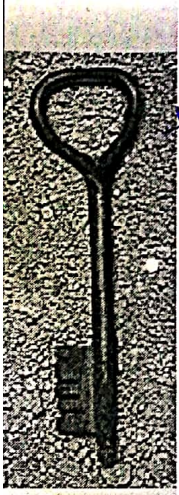
سر لی علی

Meet-in-the-Middle Attack

The use of double DES results in a mapping that is not equivalent to a single DES encryption

The meet-in-the-middle attack algorithm will attack this scheme and does not depend on any particular property of DES but will work against any block encryption cipher



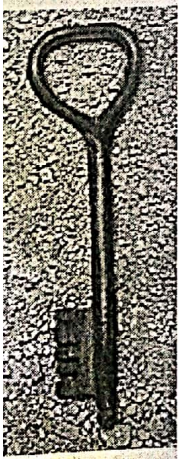


Triple-DES with Two-Keys

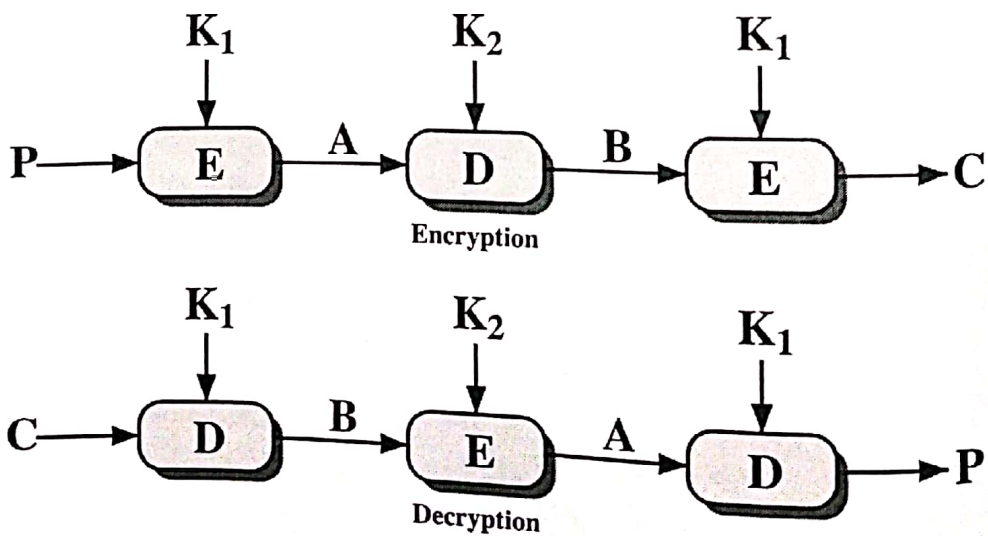
Obvious counter to the meet-in-the-middle attack is to use three stages of encryption with three different keys

- This raises the cost of the meet-in-the-middle attack to 2^{112} , which is beyond what is practical
- Has the drawback of requiring a key length of $56 \times 3 = 168$ bits, which may be somewhat unwieldy
- As an alternative Tuchman proposed a triple encryption method that uses only two keys

3DES with two keys is a relatively popular alternative to DES and has been adopted for use in the key management standards ANSI X9.17 and ISO 8732



Multiple Encryption



(b) Triple Encryption

Figure 6.1 Multiple Encryption



Triple DES with Three Keys ~~X~~

- Many researchers now feel that three-key 3DES is the preferred alternative

Three-key 3DES has an effective key length of 168 bits and is defined as:

- $C = E(K_3, D(K_2, E(K_1, P)))$

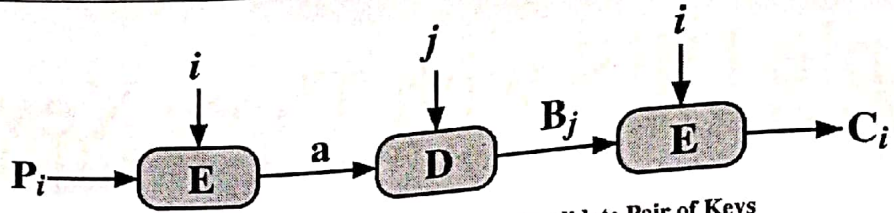
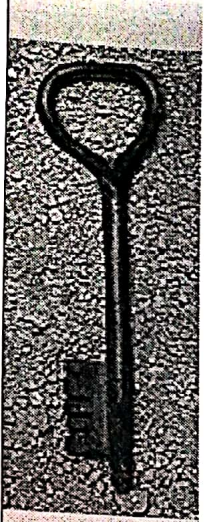
Backward compatibility with DES is provided by putting:

- $K_3 = K_2$ or $K_1 = K_2$

A number of Internet-based applications have adopted three-key 3DES including PGP and S/MIME



Next is AES



(a) Two-key Triple Encryption with Candidate Pair of Keys

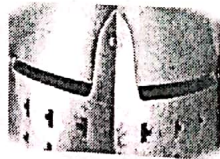
P_i	C_i

(b) Table of n known plaintext-ciphertext pairs, sorted on P

B_j	key i

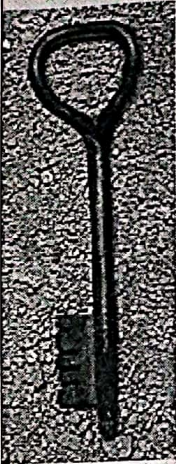
(c) Table of intermediate values and candidate keys

Figure 6.2 Known-Plaintext Attack on Triple DES



Advanced Encryption Standard

Modified by: Dr. Ramzi Saifan

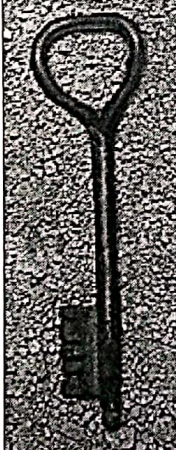


Why AES?

- ◆ Symmetric block cipher, published in 2001
- ◆ Intended to replace DES and 3DES

DES is vulnerable to multiple attacks ^{Triple}

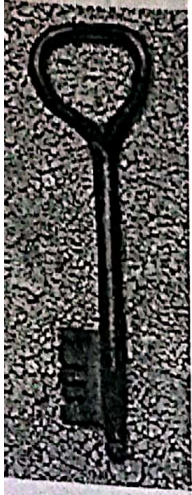
3DES has slow performances



NIST Criteria to Evaluate Potential Candidates

- ◆ Security: The effort to crypt analyze an algorithm.
- ◆ Cost: The algorithm should be practical in a wide range of applications.
- ◆ Algorithm and Implementation Characteristics : Flexibility, simplicity etc.

5 final candidates have been chosen out of 15



AES Encryption Process

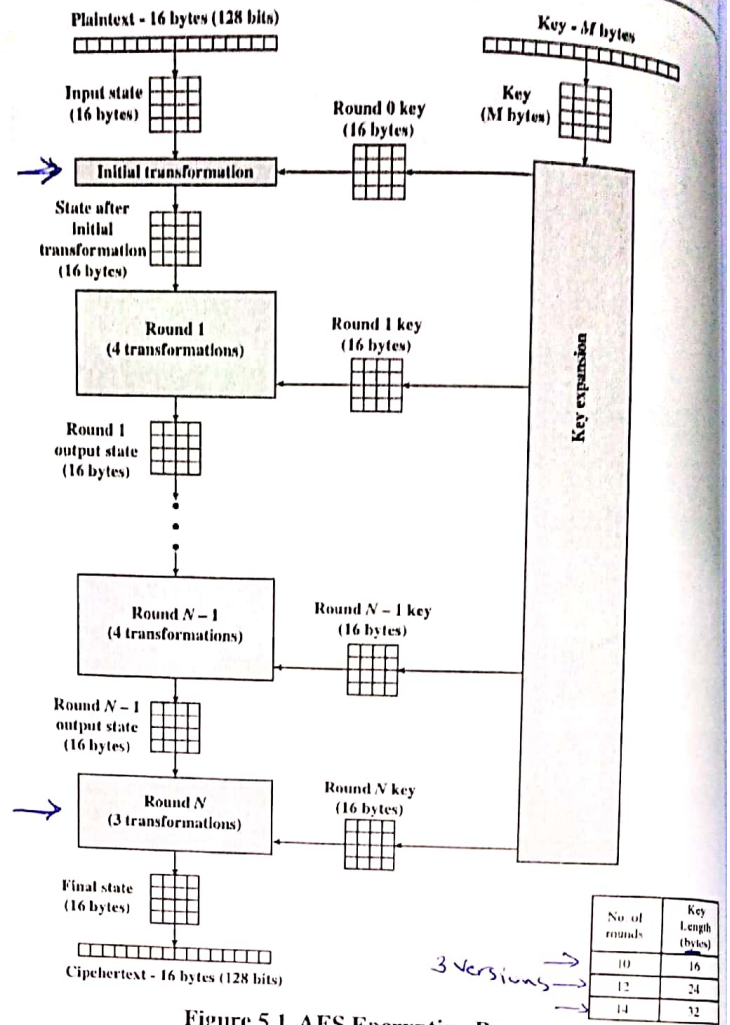
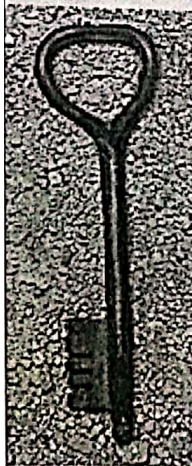
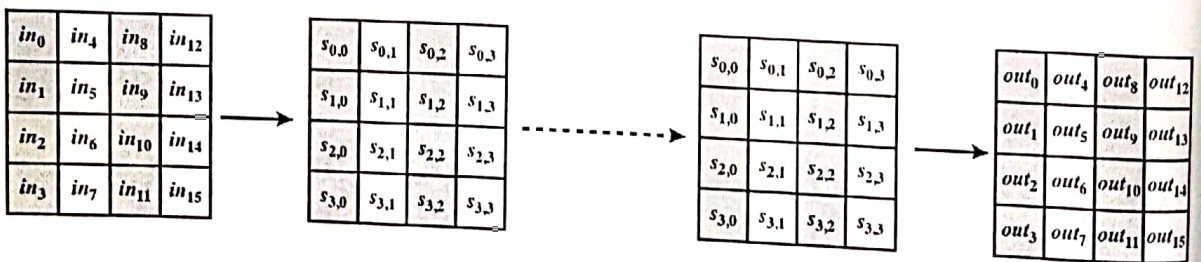


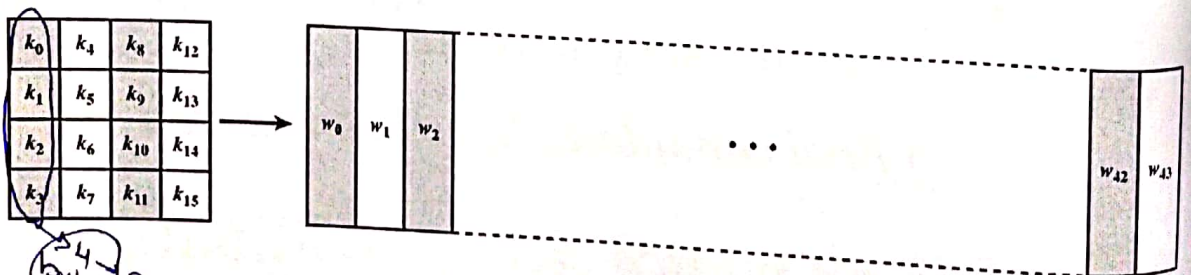
Figure 5.1 AES Encryption Process



AES Data Structures



(a) Input, state array, and output



(b) Key and expanded key



Convert to State Array

Input block:

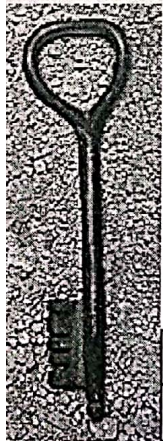
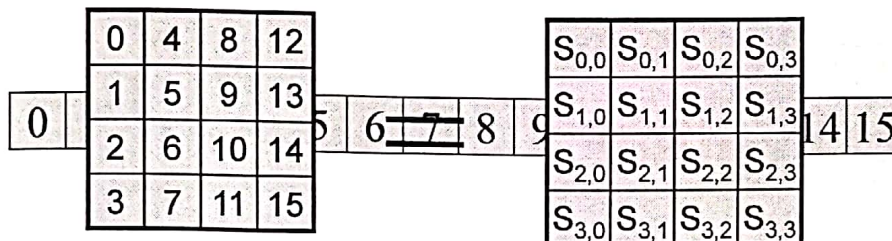
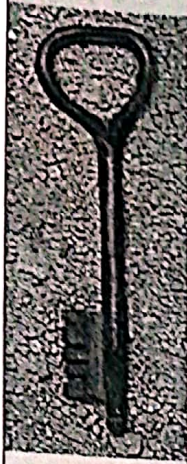


Table 5.1 AES Parameters

Key Size (words/bytes/bits)	4/16/128	6/24/192	8/32/256
Plaintext Block Size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Number of Rounds	10 = 11	12 = 13	14 = 15
Round Key Size (words/bytes/bits)	4/16/128	4/16/128	4/16/128
Expanded Key Size (words/bytes)	44/176	52/208	60/240

→ + Round 0
1



AES Encryption and Decryption

Round 0

Identical

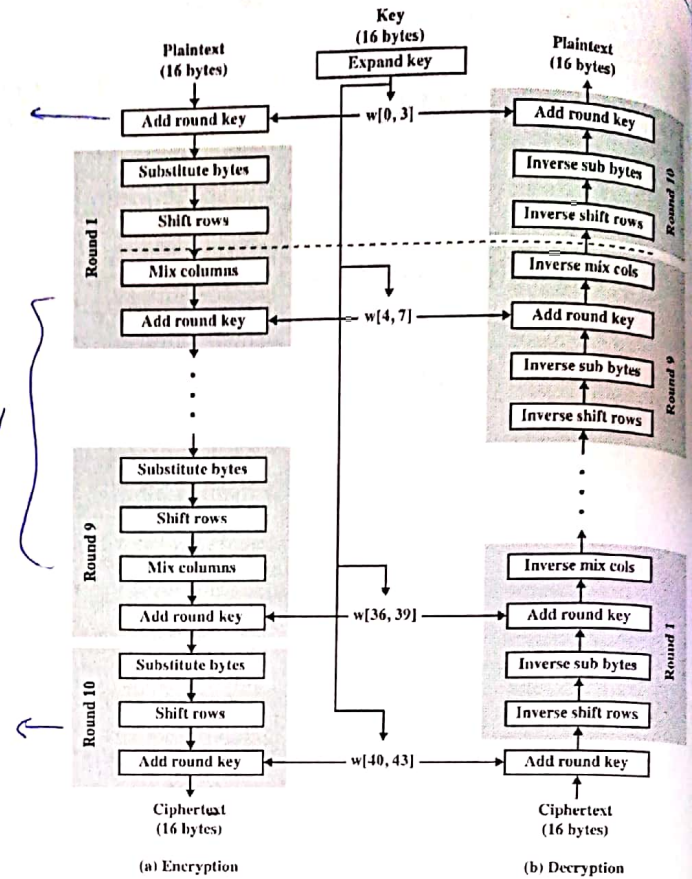
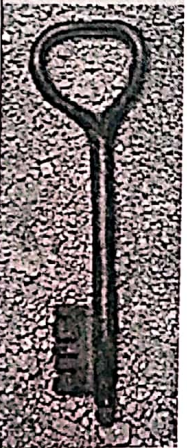


Figure 5.3 AES Encryption and Decryption



Detailed Structure

The key that is provided as input is expanded into an array of forty-four 32-bit words, $w[i]$

Four different stages are used:

- Substitute bytes – uses an S-box to perform a byte-by-byte substitution of the block
- ShiftRows – a simple permutation
- MixColumns – a substitution that makes use of arithmetic
- AddRoundKey – a simple bitwise XOR of the current block with a portion of the expanded key

Can view the cipher as alternating operations of XOR encryption (AddRoundKey) of a block, followed by scrambling of the block (the other three stages), followed by XOR encryption, and so on

Each stage is easily reversible

The decryption algorithm makes use of the expanded key in reverse order, however the decryption algorithm is not identical to the encryption algorithm

Final round of both encryption and decryption consists of only three stages

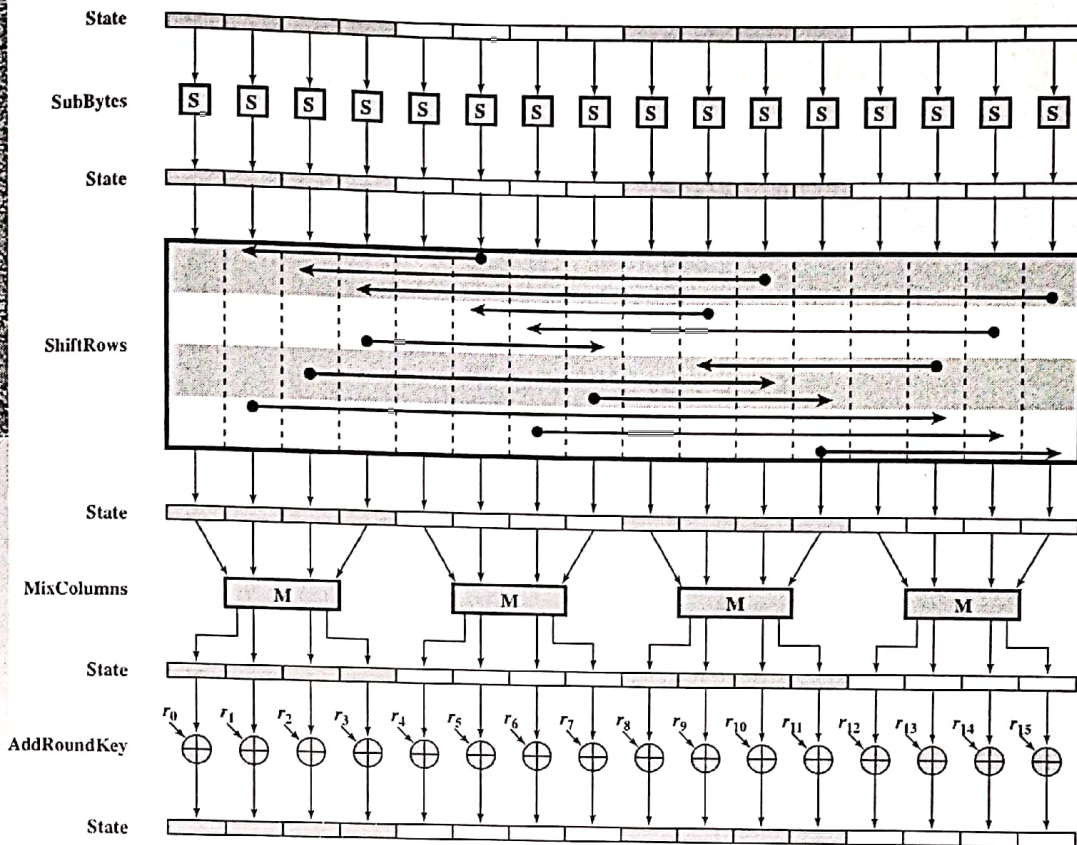
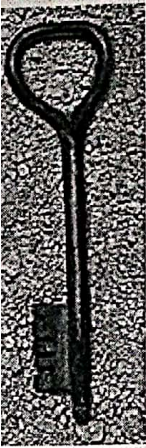
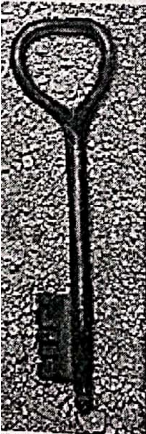


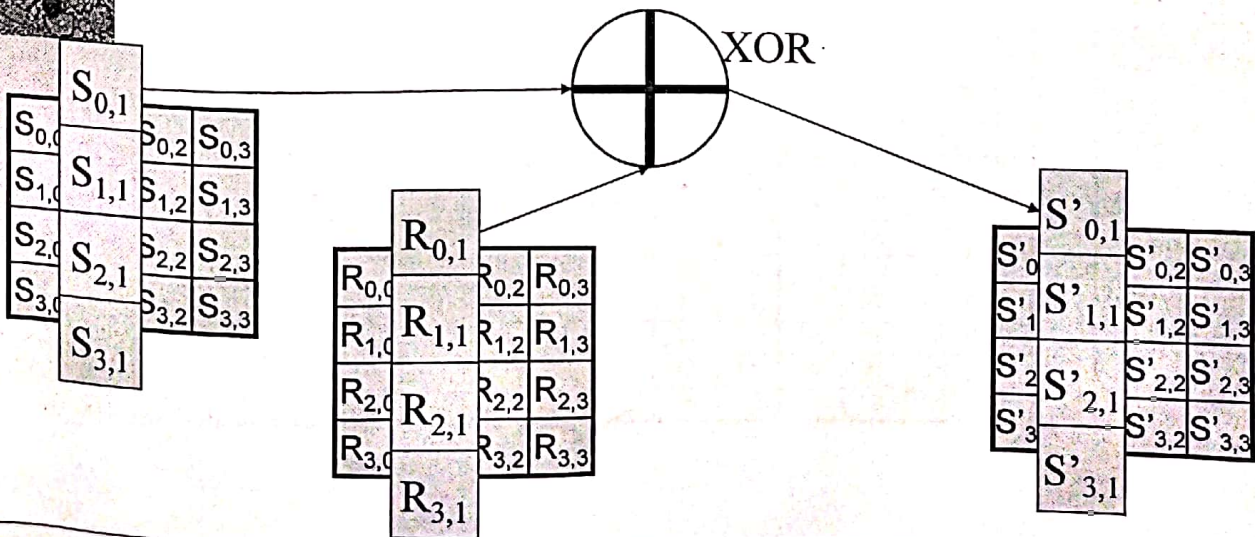
Figure 5.4 AES Encryption Round

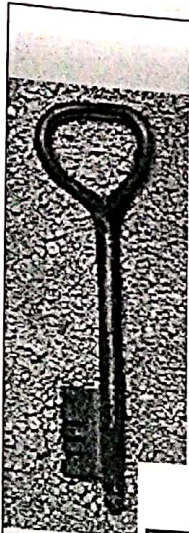


AddRoundKey

* Missed slide
 (AddRoundKey Transformation)
 (مغفلة)
 موجودة
 طاعت

- ◆ XOR each byte of the round key with its corresponding byte in the state array



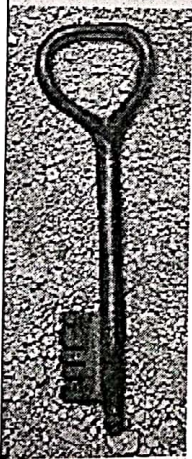


SubBytes

8-bits

- ◆ Replace each byte in the state array with its corresponding value from the S-Box

	y															
x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	7d	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16



(a) S-box

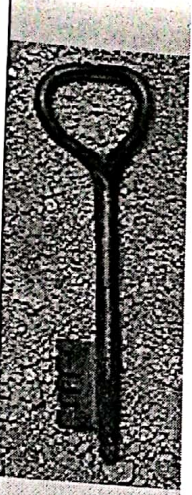
	y															
x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16

عنه ارجع، لفة، لاصدية
(b) Inverse S-box

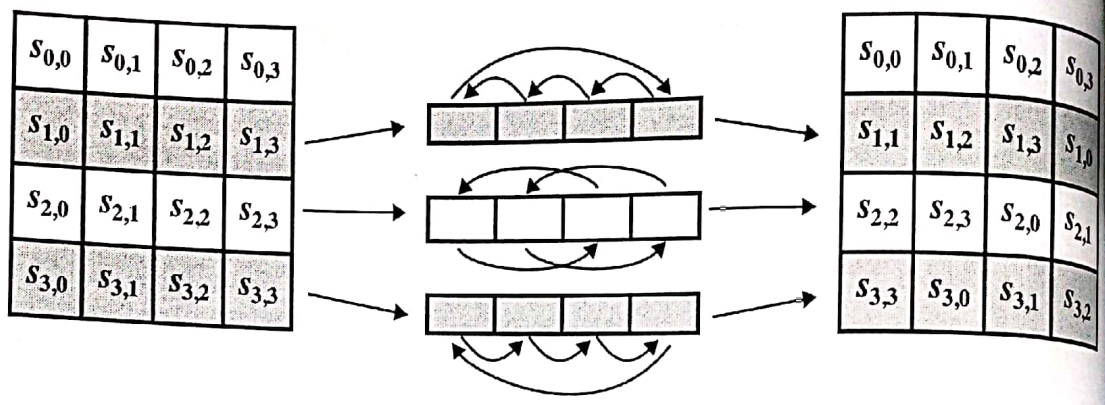
		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FB
	1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CB
	2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
	3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	DI	25
	4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
	5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
	6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
	7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	6B
	8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
	9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6E
	A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1B
	B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
	C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	5F
	D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	EF
	E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
	F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	7D

S-Box Rationale

- ◆ The S-box is designed to be resistant to known cryptanalytic attacks
- ◆ The Rijndael developers sought a design that has a low correlation between input bits and output bits and the property that the output is not a linear mathematical function of the input

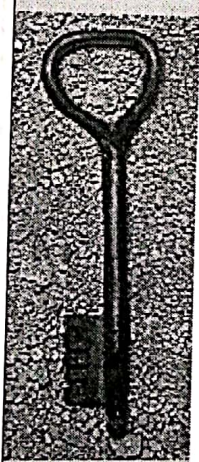


Shift Row Transformation



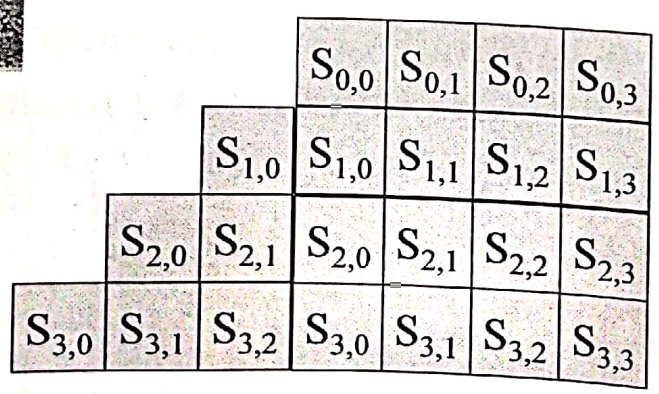
(a) Shift row transformation

AES Row and Column Operations



ShiftRows

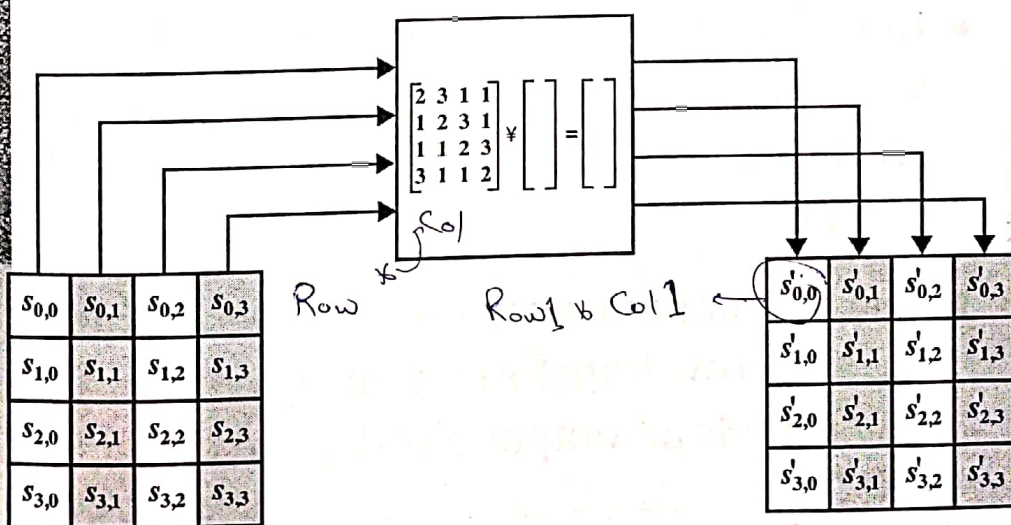
- ◆ Last three rows are cyclically shifted



Shift Row Rationale

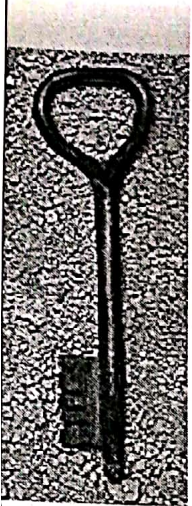
- On encryption, the first 4 bytes of the plaintext are copied to the first column of State, and so on
- The round key is applied to State column by column
 - Thus, a row shift moves an individual byte from one column to another, which is a linear distance of a multiple of 4 bytes
- Transformation ensures that the 4 bytes of one column are spread out to four different columns

MixColumn Transformation



(b) Mix column transformation
Figure 5.7 AES Row and Column Operations

(Figure can be found on page 144 in textbook)

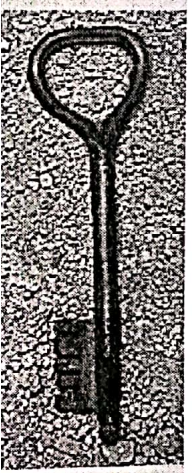


MixColumns

- ◆ Apply MixColumn transformation to each column

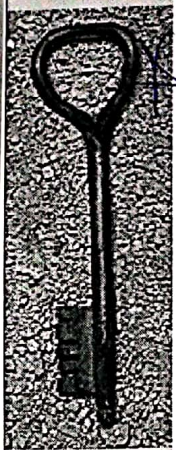
$$\begin{aligned}
 S'_{0,c} &= (\{02\} \cdot S_{0,c}) \oplus (\{03\} \cdot S_{1,c}) \oplus S_{2,c} \oplus S_{3,c} \\
 S'_{1,c} &= S_{0,c} \oplus (\{02\} \cdot S_{1,c}) \oplus (\{03\} \cdot S_{2,c}) \oplus S_{3,c} \\
 S'_{2,c} &= S_{0,c} \oplus S_{1,c} \oplus (\{02\} \cdot S_{2,c}) \oplus (\{03\} \cdot S_{3,c}) \\
 S'_{3,c} &= (\{03\} \cdot S_{0,c}) \oplus S_{1,c} \oplus S_{2,c} \oplus (\{02\} \cdot S_{3,c})
 \end{aligned}$$

S_0	0,3
S_1	1,3
S_2	2,3
S_3	3,3



Mix Columns Rationale

- ◆ Coefficients of a matrix based on a linear code with maximal distance between code words ensures a good mixing among the bytes of each column
- ◆ The mix column transformation combined with the shift row transformation ensures that after a few rounds all output bits depend on all input bits



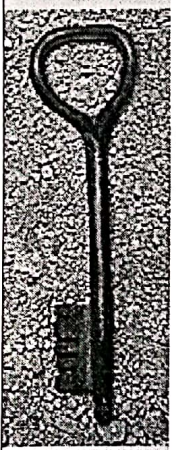
AddRoundKey Transformation

- The 128 bits of State are bitwise XORed with the 128 bits of the round key
- Operation is viewed as a columnwise operation between the 4 bytes of a State column and one word of the round key
 - Can also be viewed as a byte-level operation

Rationale:

Is as simple as possible and affects every bit of State

The complexity of the round key expansion plus the complexity of the other stages of AES ensure security



Inputs for Single AES Round

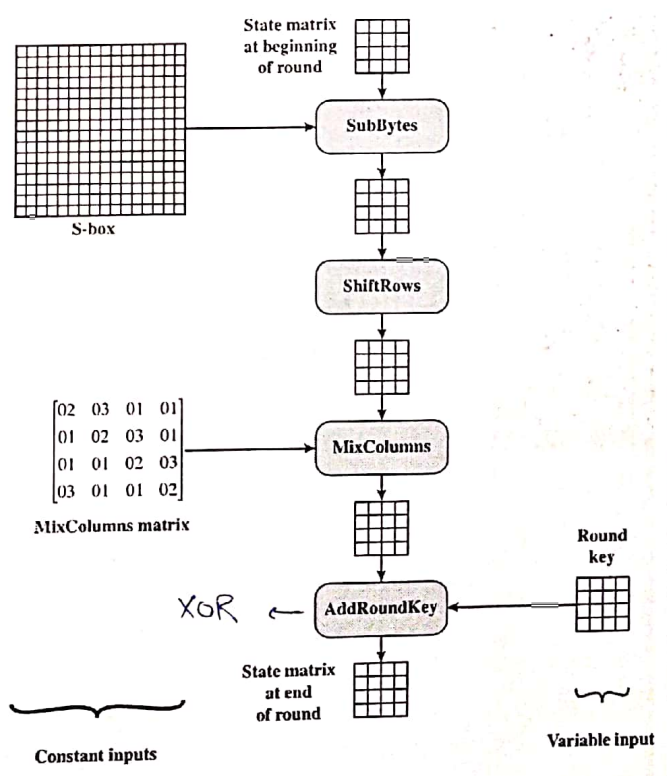


Figure 5.8 Inputs for Single AES Round

Self-Substitution
-Read

AES Key Expansion

Takes as input a four-word (16 byte) key and produces a linear array of 44 words (176) bytes

- This is sufficient to provide a four-word round key for the initial AddRoundKey stage and each of the 10 rounds of the cipher

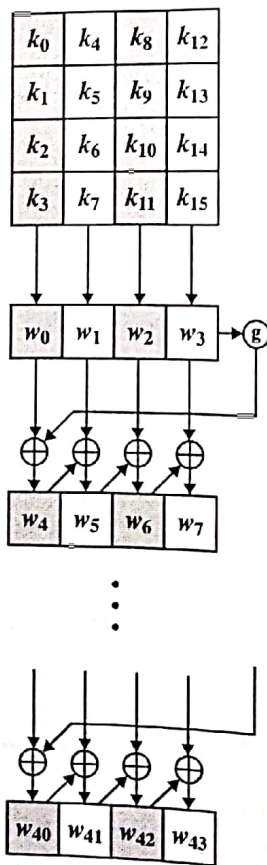
Key is copied into the first four words of the expanded key

- The remainder of the expanded key is filled in four words at a time

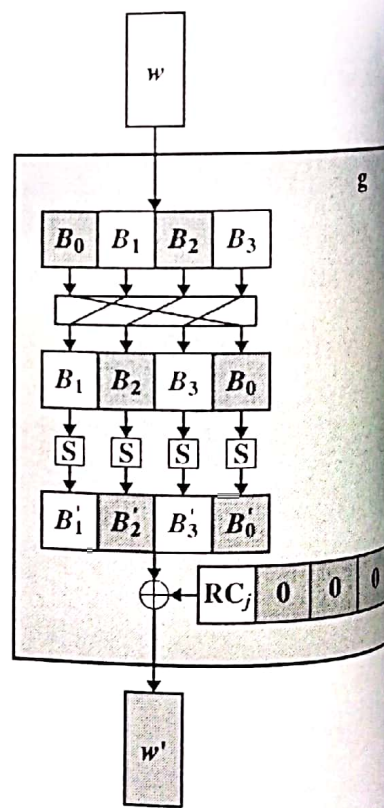
Each added word $w[i]$ depends on the immediately preceding word, $w[i - 1]$, and the word four positions back, $w[i - 4]$

- In three out of four cases a simple XOR is used
- For a word whose position in the w array is a multiple of 4, a more complex function is used

AES Key Expansion

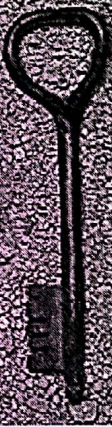


(a) Overall algorithm



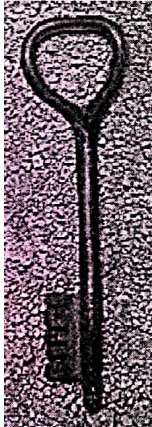
(b) Function g

Figure 5.9 AES Key Expansion



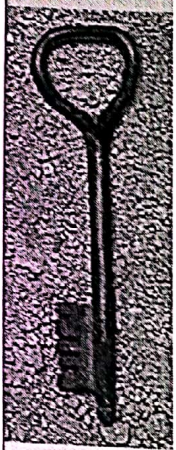
Key Expansion Rationale

- The Rijndael developers designed the expansion key algorithm to be resistant to known cryptanalytic attacks
- Inclusion of a round-dependent round constant eliminates the symmetry between the ways in which round keys are generated in different rounds



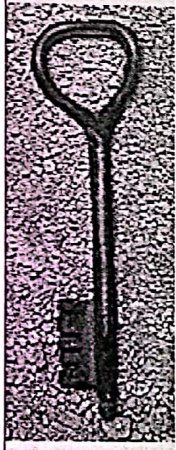
AES Example Key Expansion

Key Words	Auxiliary Function
w0 = 0f 15 71 c9 w1 = 47 d9 e8 59 w2 = 0c b7 ad d6 w3 = af 7f 67 98	RotWord(w3)= 7f 67 98 af = x1 SubWord(x1)= d2 85 46 79 = y1 Rcon(1)= 01 00 00 00 y1 ⊕ Rcon(1)= d3 85 46 79 = z1
w4 = w0 ⊕ z1 = dc 90 37 b0 w5 = w4 ⊕ w1 = 9b 49 df e9 w6 = w5 ⊕ w2 = 97 fe 72 3f w7 = w6 ⊕ w3 = 38 81 15 a7	RotWord(w7)= 81 15 a7 38 = x2 SubWord(x4)= 0c 59 5c 07 = y2 Rcon(2)= 02 00 00 00 y2 ⊕ Rcon(2)= 0e 59 5c 07 = z2
w8 = w4 ⊕ z2 = d2 c9 6b b7 w9 = w8 ⊕ w5 = 49 80 b4 5e w10 = w9 ⊕ w6 = de 7e c6 61 w11 = w10 ⊕ w7 = e6 ff d3 c6	RotWord(w11)= ff d3 c6 e6 = x3 SubWord(x2)= 16 66 b4 8e = y3 Rcon(3)= 04 00 00 00 y3 ⊕ Rcon(3)= 12 66 b4 8e = z3
w12 = w8 ⊕ z3 = c0 af df 39 w13 = w12 ⊕ w9 = 89 2f 6b 67 w14 = w13 ⊕ w10 = 57 51 ad 06 w15 = w14 ⊕ w11 = b1 ae 7e c0	RotWord(w15)= ae 7e c0 b1 = x4 SubWord(x3)= e4 f3 ba c8 = y4 Rcon(4)= 08 00 00 00 y4 ⊕ Rcon(4)= ec f3 ba c8 = 4
w16 = w12 ⊕ z4 = 2c 5c 65 f1 w17 = w16 ⊕ w13 = a5 73 0e 96 w18 = w17 ⊕ w14 = f2 22 a3 90 w19 = w18 ⊕ w15 = 43 8c dd 50	RotWord(w19)= 8c dd 50 43 = x5 SubWord(x4)= 64 c1 53 1a = y5 Rcon(5)= 10 00 00 00 y5 ⊕ Rcon(5)= 74 c1 53 1a = z5
w20 = w16 ⊕ z5 = 58 9d 36 eb w21 = w20 ⊕ w17 = fd ee 38 7d w22 = w21 ⊕ w18 = 0f cc 9b ed w23 = w22 ⊕ w19 = 4c 40 46 bd	RotWord(w23)= 40 46 bd 4c = x6 SubWord(x5)= 09 5a 7a 29 = y6 Rcon(6)= 20 00 00 00 y6 ⊕ Rcon(6)= 29 5a 7a 29 = z6
w24 = w20 ⊕ z6 = 71 c7 4c c2 w25 = w24 ⊕ w21 = 8c 29 74 bf w26 = w25 ⊕ w22 = 83 e5 ef 52 w27 = w26 ⊕ w23 = cf a5 a9 ef	RotWord(w27)= a5 a9 ef cf = x7 SubWord(x6)= 06 d3 df 8a = y7 Rcon(7)= 40 00 00 00 y7 ⊕ Rcon(7)= 46 d3 df 8a = z7
w28 = w24 ⊕ z7 = 37 14 93 48 w29 = w28 ⊕ w25 = bb 3d e7 f7 w30 = w29 ⊕ w26 = 38 d8 08 a5 w31 = w30 ⊕ w27 = f7 7d a1 4a	RotWord(w31)= 7d a1 4a f7 = x8 SubWord(x7)= ff 32 d6 68 = y8 Rcon(8)= 80 00 00 00 y8 ⊕ Rcon(8)= 7f 32 d6 68 = z8
w32 = w28 ⊕ z8 = 48 26 45 20 w33 = w32 ⊕ w29 = f3 1b a2 d7 w34 = w33 ⊕ w30 = cb c3 aa 72 w35 = w34 ⊕ w32 = 3c be 0b 38	RotWord(w35)= be 0b 38 3c = x9 SubWord(x8)= ae 2b 07 eb = y9 Rcon(9)= 1B 00 00 00 y9 ⊕ Rcon(9)= b5 2b 07 eb = z9
w36 = w32 ⊕ z9 = fd 0d 42 cb w37 = w36 ⊕ w33 = 0e 16 e0 1c w38 = w37 ⊕ w34 = c5 d5 4a 6e w39 = w38 ⊕ w35 = f9 6b 41 56	RotWord(w39)= 6b 41 56 f9 = x10 SubWord(x9)= 7f 83 b1 99 = y10 Rcon(10)= 36 00 00 00 y10 ⊕ Rcon(10)= 49 83 b1 99 = z10
w40 = w36 ⊕ z10 = b4 8e f3 52 w41 = w40 ⊕ w37 = ba 98 13 4e w42 = w41 ⊕ w38 = 7f 4d 59 20 w43 = w42 ⊕ w39 = 86 26 18 76	



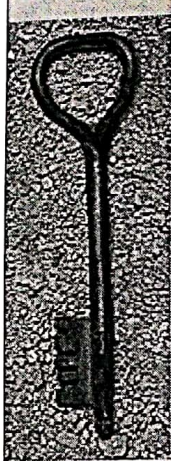
AES Example

Start of round	After SubBytes	After ShiftRows	After MixColumns	Round Key
01 89 fe 76				0f 47 0c 8f
23 ab dc 54				15 d9 b7 7e
45 cd ba 32				71 e8 ad 6f
67 ef 98 10				c9 59 d6 98
0e ce f2 d9	ab 8b 89 35	ab 8b 89 35	b9 94 57 75	dc 9b 97 38
36 72 6b 2b	05 40 7f f1	40 7f f1 05	e4 8e 16 51	90 49 fe 01
34 25 17 55	18 3f f0 fc	f0 fc 18 3f	47 20 9a 3f	37 df 72 15
ae b6 4e 88	e4 4e 2f c4	c4 e4 4e 2f	c5 d6 f5 3b	b0 e9 3f a7
65 0f c0 4d	4d 76 ba e3	4d 76 ba e3	8e 22 db 12	d2 49 de ee
74 c7 e8 d0	92 c6 9b 70	c6 9b 70 92	b2 f2 dc 92	c9 80 7e ff
70 ff e8 2a	51 16 9b e5	9b e5 16 92	df 80 f7 c1	6b b4 ce d3
75 3f ca 9c	9d 75 74 de	de 9d 75 74	2d c5 1e 52	b7 5e 61 ce
5c 6b 05 f4	4a 7f 6b bf	4a 7f 6b bf	b1 c1 0b cc	c0 89 57 01
7b 72 a2 6d	21 40 3a 3c	40 3a 3c 21	ba f3 8b 07	af 2f 51 ae
b4 34 31 12	8d 18 c7 c9	c7 c9 8d 18	f9 1f 6a c3	df 6b ad 7e
9a 9b 7f 94	b8 14 d2 22	22 b8 14 d2	1d 19 24 5c	39 67 06 c0
71 48 5c 7d	a3 52 4a ff	a3 52 4a ff	d4 11 fe 0f	2c a5 f2 43
15 cd da a9	59 86 57 d3	86 57 d3 59	3b 44 06 73	5c 73 22 8c
26 74 c7 bd	f7 92 c6 7a	c6 7a f7 92	cb ab 62 37	65 0e a3 dd
24 7e 22 9c	36 f3 93 de	de 36 f3 93	19 b7 07 ec	f1 96 90 50
f8 b4 0c 4c	41 8d fe 29	41 8d fe 29	2a 47 c4 48	58 fd 0f 4c
67 37 24 ff	85 9a 36 16	9a 36 16 85	83 e8 18 ba	9d ee cc 40
ae a5 c1 ea	e4 06 78 87	78 87 e4 06	84 18 27 23	36 38 9b 46
e8 21 97 bc	9b fd 88 65	65 9b fd 88	eb 10 0a f3	eb 7d ed bd
72 ba cb 04	40 f4 1f f2	40 f4 1f f2	7b 05 42 4a	71 8c 83 cf
1e 06 d4 fa	72 6f 48 2d	6f 48 2d 72	1e d0 20 40	c7 29 e5 a5
b2 20 bc 65	37 b7 65 4d	65 4d 37 b7	94 83 18 52	4c 74 ef a9
00 6d e7 4e	63 3c 94 2f	2f 63 3c 94	94 c4 43 fb	c2 bf 52 ef
0a 89 c1 85	67 a7 78 97	67 a7 78 97	ec 1a c0 80	37 bb 38 f7
d9 f9 c5 e5	35 99 a6 d9	99 a6 d9 35	0c 50 53 c7	14 3d d8 7d
d8 f7 f7 fb	61 68 68 0f	68 0f 61 68	3b d7 00 ef	93 e7 08 al
56 7b 11 14	b1 21 82 fa	fa b1 21 82	b7 22 72 e0	48 f7 a5 4a
db a1 f8 77	b9 32 41 f5	b9 32 41 f5	b1 1a 44 17	48 f3 cb 3c
18 6d 8b ba	ad 3c 3d f4	3c 3d f4 ad	3d 2f ec b6	26 1b c3 be
a8 30 08 4e	c2 04 30 2f	30 2f c2 04	0a 6b 2f 42	45 a2 aa 0b
ff d5 d7 aa	16 03 0e ac	ac 16 03 0e	9f 68 f3 b1	20 d7 72 38
f9 e9 8f 2b	99 1e 73 f1	99 1e 73 f1	31 30 3a c2	fd 0e c5 f9
1b 34 2f 08	af 18 15 30	18 15 30 af	ac 71 8c c4	0d 16 d5 6b
4f c9 85 49	84 dd 97 3b	97 3b 84 dd	46 65 48 eb	42 e0 4a 41
bf bf 81 89	08 08 0c a7	a7 08 08 0c	6a 1c 31 62	cb 1c 6e 56
cc 3e ff 3b	4b b2 16 e2	4b b2 16 e2	4b 86 8a 36	b4 ba 7f 86
a1 67 59 af	32 85 cb 79	85 cb 79 32	b1 cb 27 5a	8e 98 4d 26
04 85 02 aa	f2 97 77 ac	77 ac f2 97	fb f2 f2 af	f3 13 59 18
a1 00 5f 34	32 63 cf 18	18 32 63 cf	cc 5a 5b cf	52 4e 20 76
ff 08 69 64				
0b 53 34 14				
84 bf ab 8f				
4a 7c 43 b9				



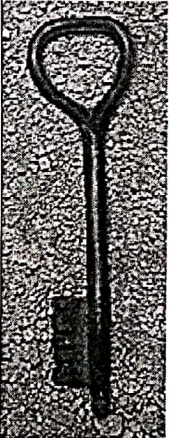
Avalanche Effect in AES: Change in Plaintext

Round	Number of Bits that Differ
0	1
1	20
2	58
3	59
4	61
5	68
6	64
7	67
8	65
9	61
10	58



Avalanche
Effect
in AES:
Change
in Key

Round		Number of Bits that Differ
	0123456789abcdef fedcba9876543210 0123456789abcdef fedcba9876543210	0
0	0e3634aece7225b6f26b174ed92b5588 0f3634aece7225b6f26b174ed92b5588	1
1	657470750fc7ff3fc0e8e8ca4dd02a9c c5a9ad090ec7ff3fc1e8e8ca4cd02a9c	22
2	5c7bb49a6b72349b05a2317ff46d1294 90905fa9563356d15f3760f3b8259985	58
3	7115262448dc747e5cdac7227da9bd9c 18aeb7aa794b3b66629448d575c7cebf	67
4	f867aee8b437a5210c24c1974cffeabc f81015f993c978a876ae017cb49e7eec	63
5	721eb200ba06206dcbd4bce704fa654e 5955c91b4e769f3cb4a94768e98d5267	81
6	0ad9d85689f9f77bc1c5f71185e5fb14 dc60a24d137662181e45b8d3726b2920	70
7	db18a8ffa16d30d5f88b08d777ba4eaa fe8343b8f88bef66cab7e977d005a03c	74
8	f91b4fbfe934c9bf8f2f85812b084989 da7dad581d1725c5b72fa0f9d9d1366a	67
9	cca104a13e678500ff59025f3bafaa34 0ccb4c66bbfd912f4b511d72996345e0	59
10	ff0b844a0853bf7c6934ab4364148fb9 fc8923ee501a7d207ab670686839996b	53

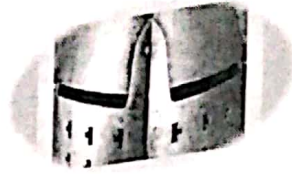


Implementation Aspects

- AddRoundKey is a bitwise XOR operation
- ShiftRows is a simple byte-shifting operation
- SubBytes operates at the byte level and only requires a table of 256 bytes
- MixColumns requires matrix multiplication
- MixColumns only requires multiplication by {02} and {03}, which can be converted to shifts and XORs.
- Designers believe this very efficient implementation was a key factor in its selection as the AES cipher.

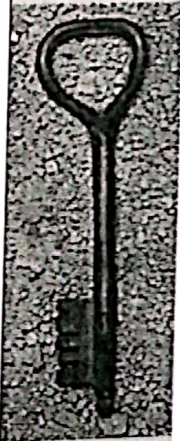
Note:

DES, AES → are block cipher



*Modes of Operation

Modified by: Dr. Ramzi Saifan



Modes of Operation

To apply a block cipher in a variety of applications, five modes of operation have been defined by NIST.

- The five modes are intended to cover a wide variety of applications of encryption for which a block cipher could be used
- These modes are intended for use with any symmetric block cipher, including triple DES and AES

divide plain text into blocks

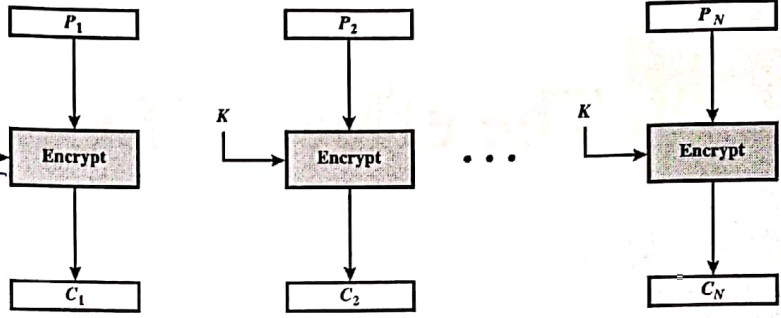
disadv (cons)

- * Same blocks have same cipher text
- * Last block maybe padding

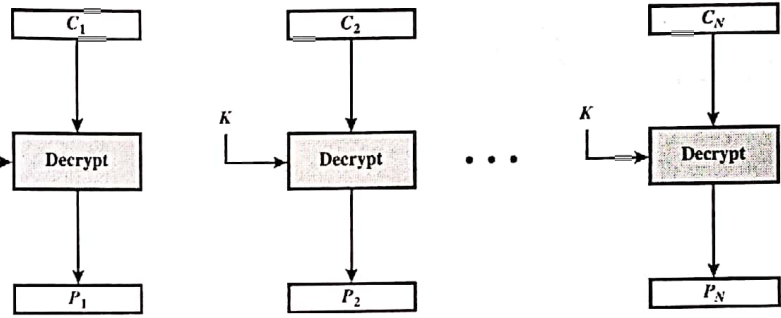
Electronic Codebook Mode (ECB)

adv (pros)

- * error doesn't propagate
- * parallel Enc/Dec
- * Faster



(a) Encryption

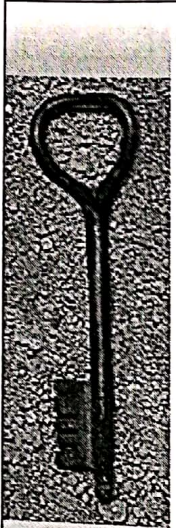


(b) Decryption

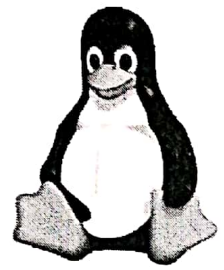
$$C_i = E_K(P_i); \text{ the cipher text is } (C_1, \dots, C_n)$$

Security?

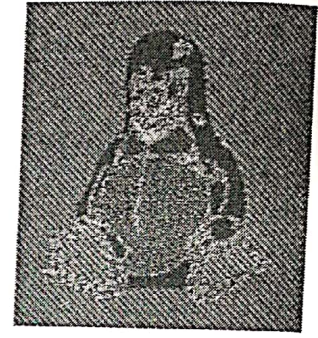
- ◆ ECB should *not* be used until the message is small
 - Why? to avoid repetition.



The effect of ECB mode



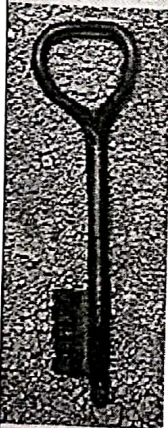
original



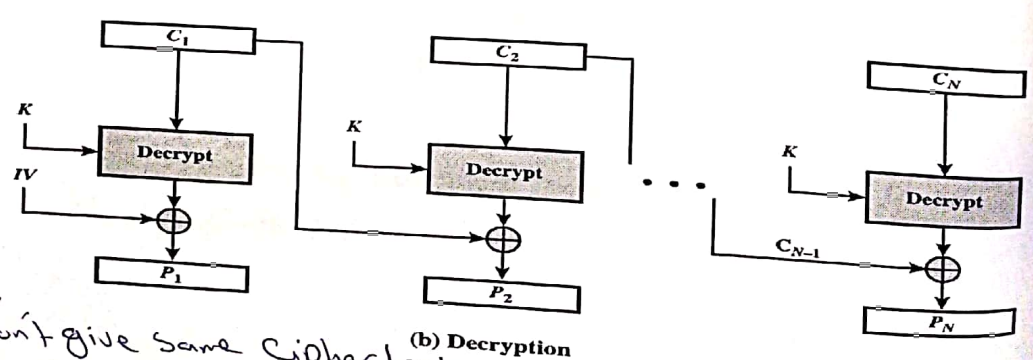
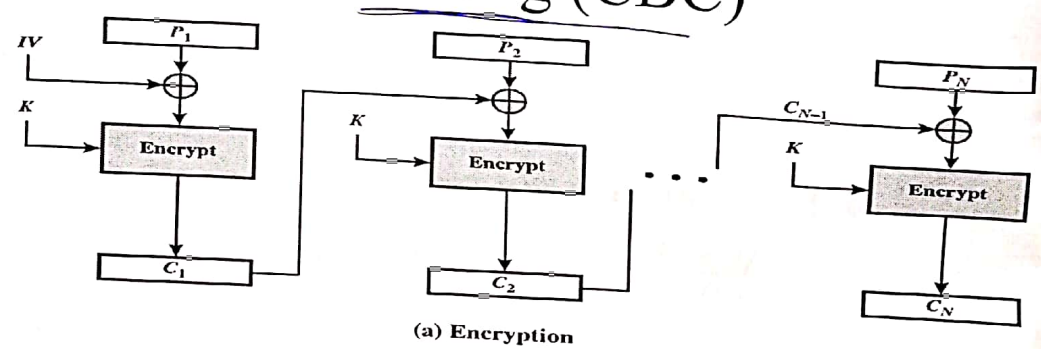
encrypted using ECB mode

Same blocks encrypted into same cipher text.

*Images from Wikipedia



Cipher Block Chaining (CBC)



* loss
 * slow
 * error is propagated
 * last block padding.

* Same blocks won't give same cipher text.

$$IV; C_i = E_K(m_i \oplus C_{i-1}); \text{ the ciphertext is } (IV, C_1, \dots, C_N)$$

Cipher Feedback Mode

- ◆ For AES, DES, or any block cipher, encryption is performed on a block of b bits

- In the case of DES $b=64$
- In the case of AES $b=128$

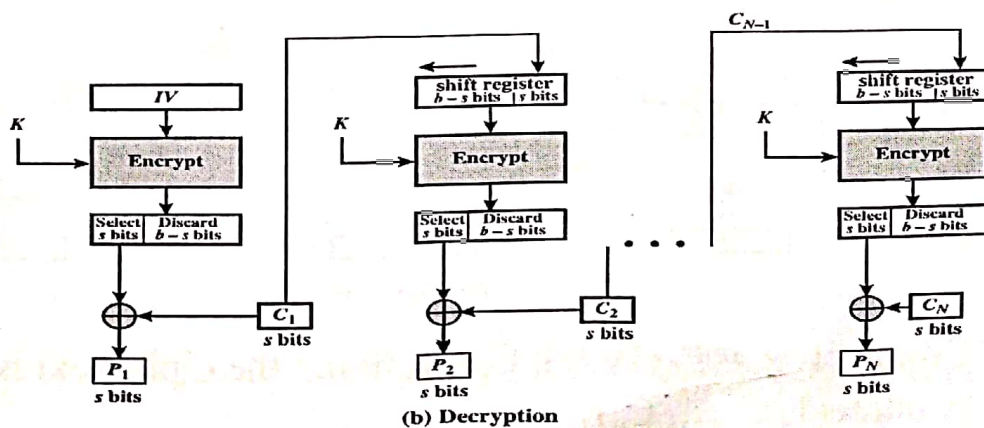
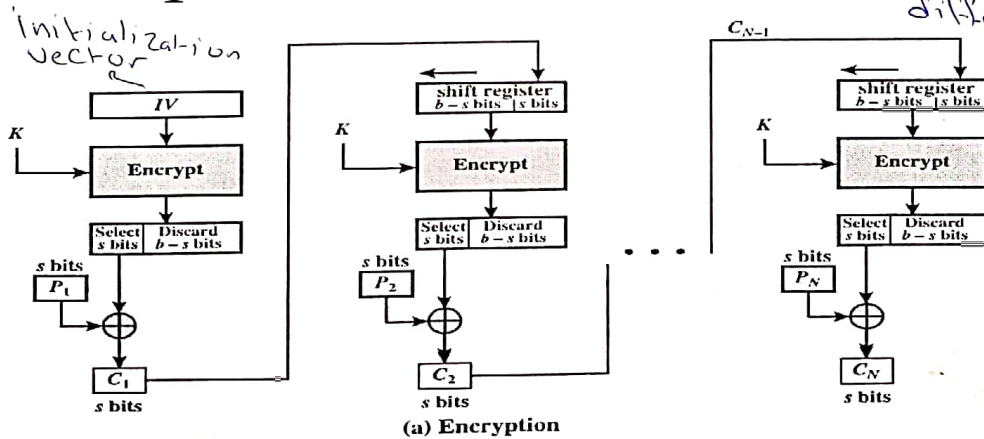
There are three modes that make it possible to convert a block cipher into a stream cipher:

Cipher feedback (CFB) mode

Output feedback (OFB) mode

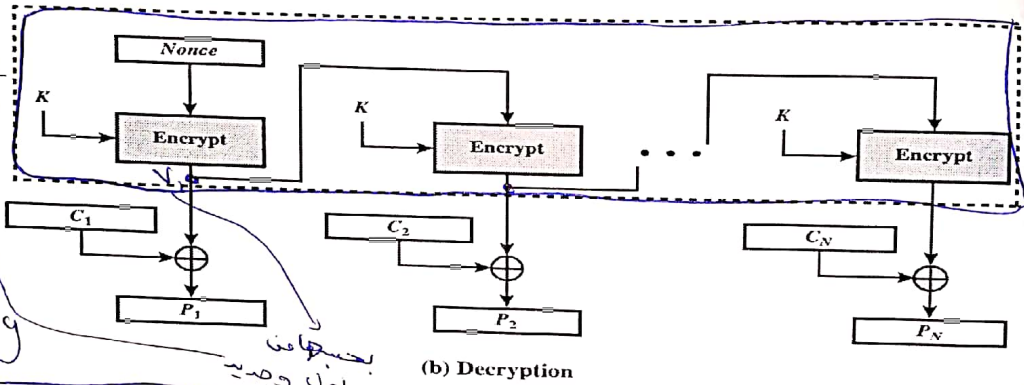
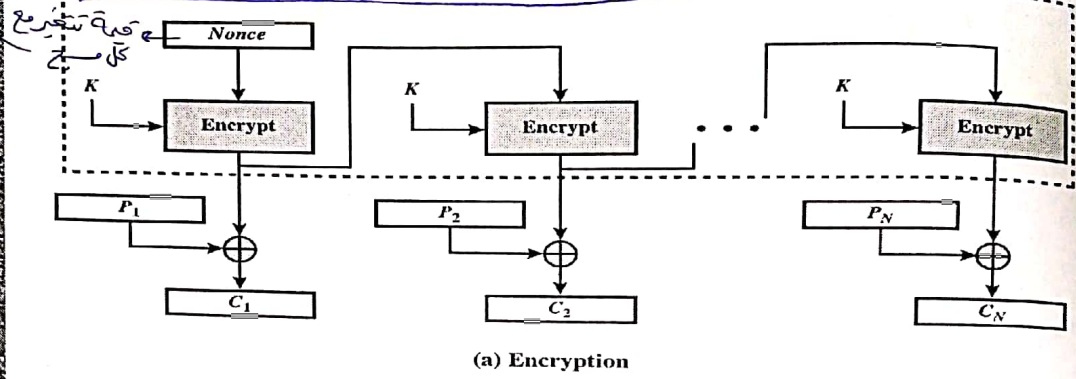
Counter (CTR) mode

s -bit Cipher Feedback (CFB) Mode



* No padding
* same blocks
different cipher
* No parallel
* Error propagate

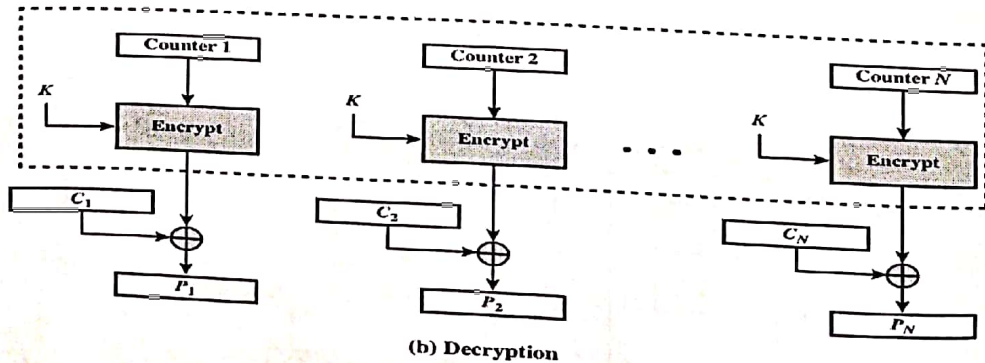
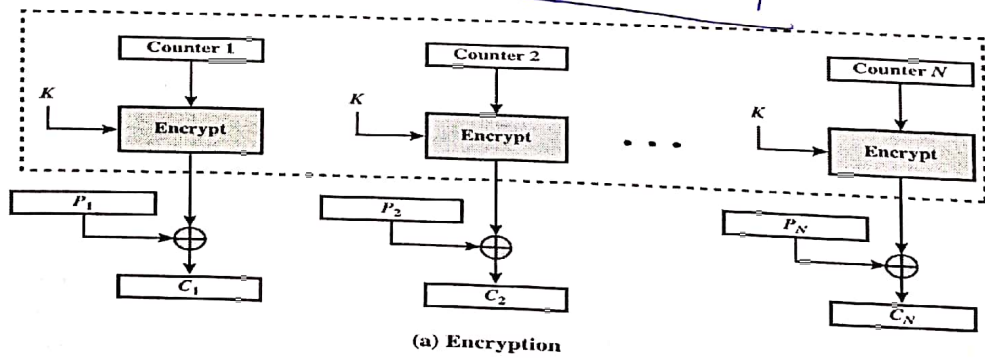
Output Feedback (OFB) Mode



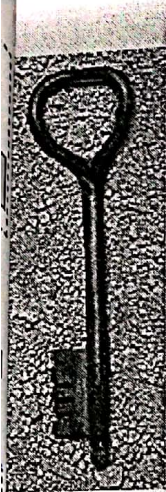
* Same blocks won't have same cipher
 * No padding
 * Error doesn't propagate
 * Preprocessing
 * بضعه اول و بعد

Nonce; $z_i = E_K(z_{i-1})$; $C_i = z_i \oplus m_i$; the ciphertext is (Nonce, C_1, \dots, C_N)

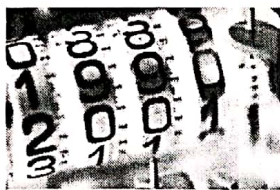
Counter (CTR) Mode



Counter 1; $z_i = F_K(IV+i)$; $C_i = z_i \oplus m_i$; the ciphertext is (Counter 1, C_1, \dots, C_N)

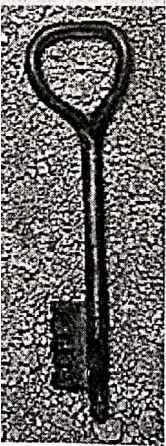


Advantages of CTR



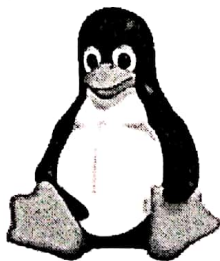
- ◆ Hardware efficiency
- ◆ Software efficiency
- ◆ Preprocessing
- ◆ Random access
- ◆ Provable security
- ◆ Simplicity

- * Parallel
- * Fast
- * Error isn't propagated.
- * No padding (Stream Cipher)



Security

- ◆ CBC, OFB, and CTR modes are secure against chosen-plaintext attacks

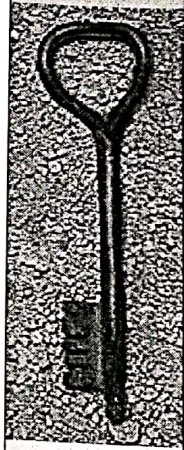


Same blocks will have different cipher text.

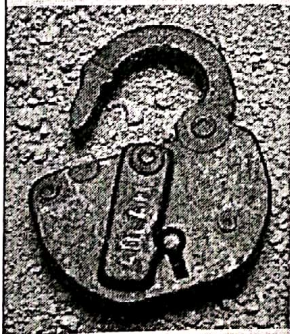


*Images from Wikipedia

Table 6.1 Block Cipher Modes of Operation

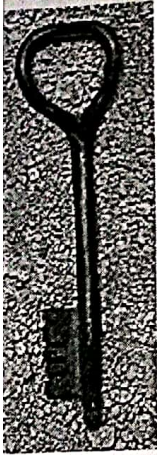


Mode	Description	Typical Application
Electronic Codebook (ECB)	Each block of plaintext bits is encoded independently using the same key.	•Secure transmission of single values (e.g., an encryption key)
Cipher Block Chaining (CBC)	The input to the encryption algorithm is the XOR of the next block of plaintext and the preceding block of ciphertext.	•General-purpose block-oriented transmission •Authentication
Cipher Feedback (CFB)	Input is processed s bits at a time. Preceding ciphertext is used as input to the encryption algorithm to produce pseudorandom output, which is XORed with plaintext to produce next unit of ciphertext.	•General-purpose stream-oriented transmission •Authentication
Output Feedback (OFB)	Similar to CFB, except that the input to the encryption algorithm is the preceding encryption output, and full blocks are used.	•Stream-oriented transmission over noisy channel (e.g., satellite communication)
Counter (CTR)	Each block of plaintext is XORed with an encrypted counter. The counter is incremented for each subsequent block.	•General-purpose block-oriented transmission •Useful for high-speed requirements



* Data Integrity

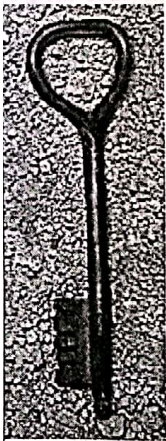
Modified by: Dr. Ramzi Saifan



Encryption/Decryption

- ◆ Provides message confidentiality.
- ◆ Does it provide message authentication?

2



Message Authentication

- Bob receives a message m from Alice, he wants to know
 - (Data origin authentication) whether the message was really sent by Alice;
 - (Data integrity) whether the message has been modified.

Solutions:

- Alice attaches a message authentication code (MAC) to the message.

Message \rightarrow $\begin{matrix} \text{Func} \\ \downarrow \\ \text{Key} \end{matrix}$ \rightarrow MAC

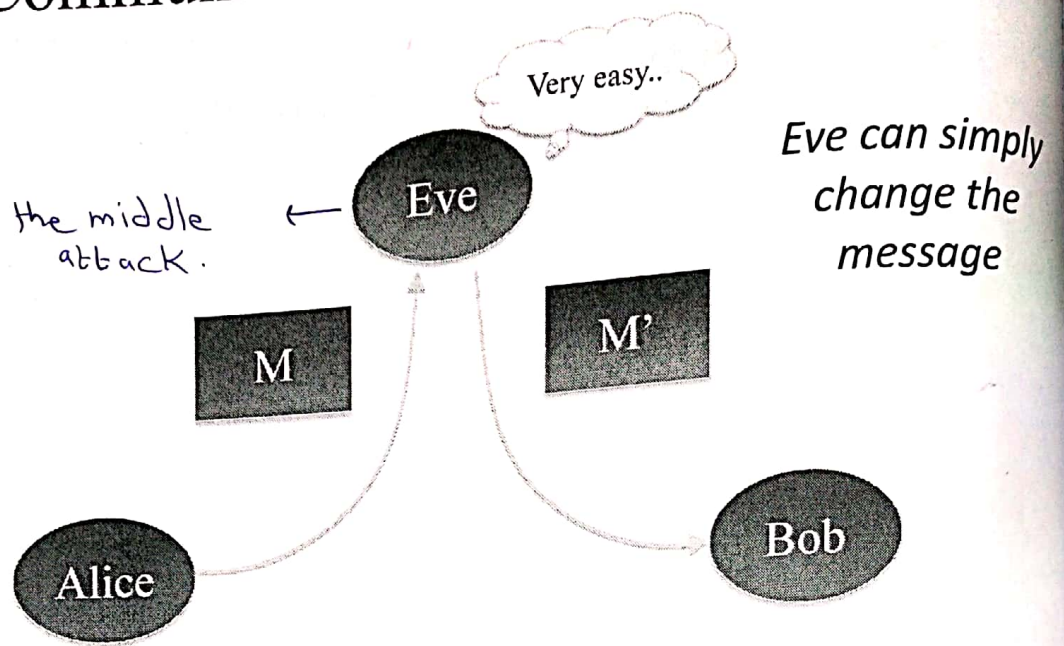
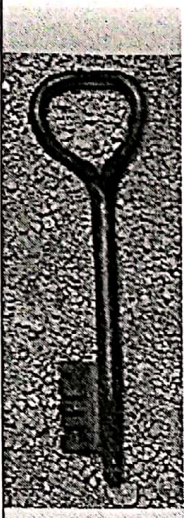
- Or she attaches a digital signature to the message.

\hookrightarrow in public key.

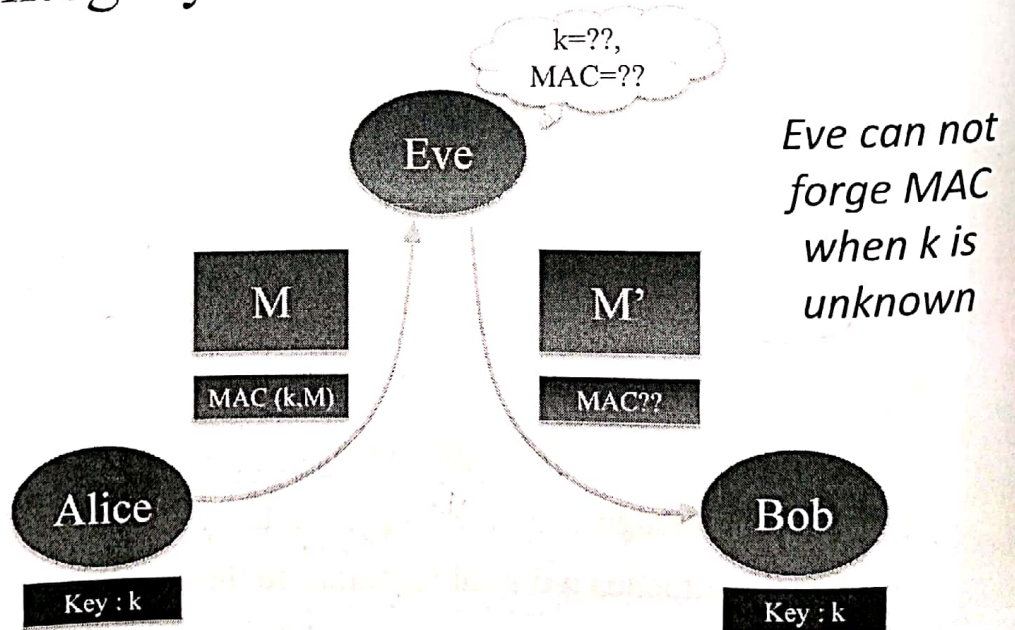
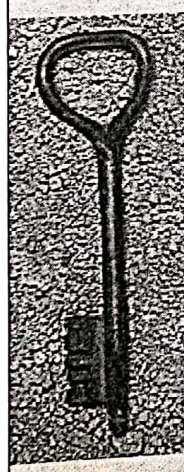
Send the MAC with the Message.

3

Communication without authentication



Integrity Protection with MAC



Shared key k to generate authenticate message

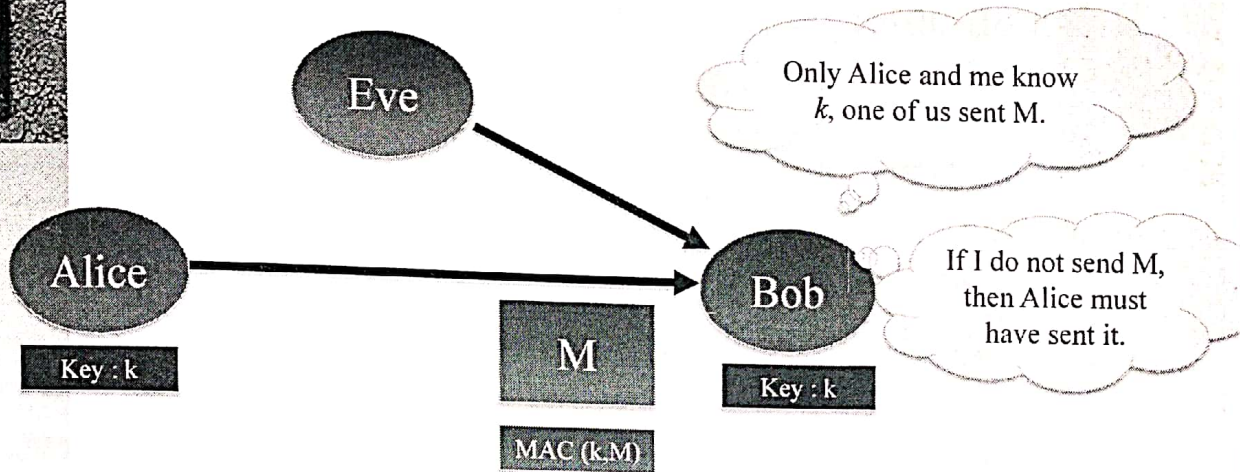
to calculate MAC 2 ways

- Hash Function (HMAC)
- Cipher techniques (CMAC)



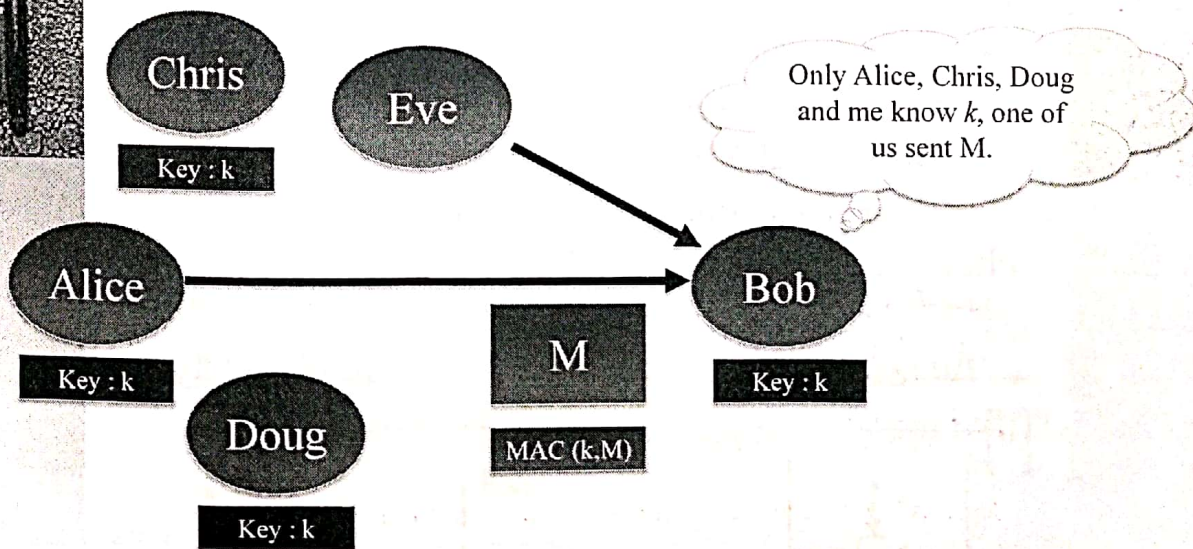
MAC Authentication (I)

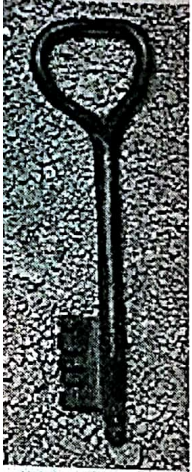
- ◆ MAC allows **two** or more mutually trusting parties to authenticate messages sent between members



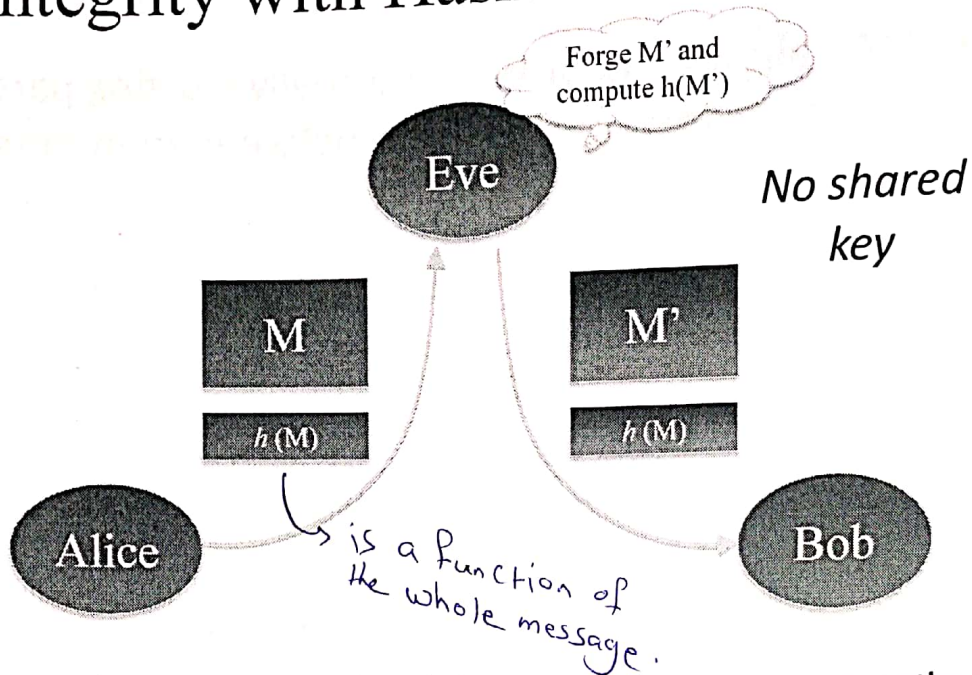
MAC Authentication (II)

- ◆ MAC allows two or **more** mutually trusting parties to authenticate messages sent between members





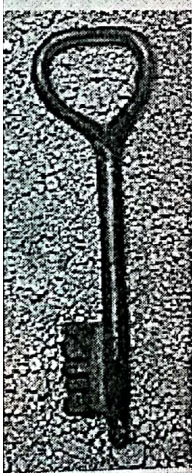
Integrity with Hash



Can we simply send the hash with the message to serve message authentication ?

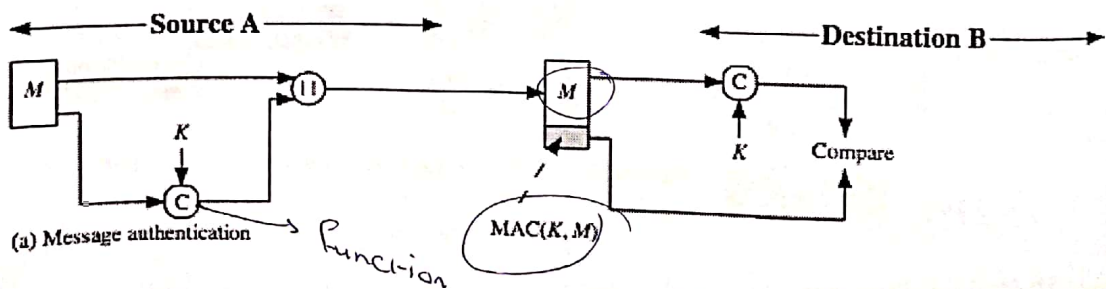
Ans: No, Eve can change the message and recompute the hash.

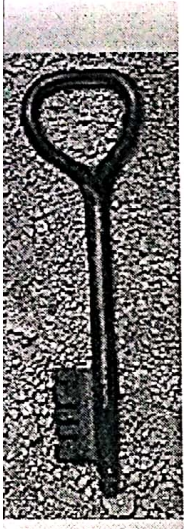
Using hash needs more appropriate procedure to guarantee integrity



Message Authentication Code

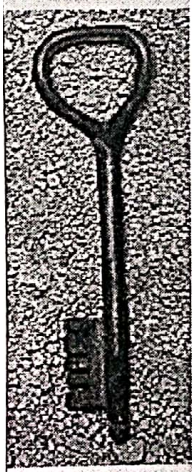
- A function of the message and a secret key that produces a fixed-length value that serves as the authenticator
- Generated by an algorithm :
 - generated from message + secret key : $MAC = F(K, M)$
 - A small fixed-sized block of data
 - appended to message as a **signature** when sent
- Receiver performs same computation on message and checks it matches the MAC





MAC and Encryption

- As shown the MAC provides authentication
- But encryption can also provides authentication!
- Why use a MAC?
 - sometimes **only authentication is needed**
 - sometimes need authentication to persist longer than the encryption (eg. archival use)



MAC Properties

- A MAC is a cryptographic hash

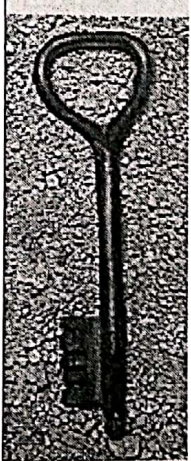
$$\text{MAC} = C_K(M)$$

- condenses a variable-length message M
 - using a secret key K
 - to a fixed-sized authenticator
- A many-to-one function
 - potentially many messages have same MAC
 - but finding these needs to be very difficult

example in the video
 week 5, lecture of
 9. Nov

	Bit 1	Bit 2	...	Bit n
Block 1	b_{11}	b_{21}		b_{n1}
Block 2	b_{12}	b_{22}		b_{n2}
	•	•	•	•
	•	•	•	•
	•	•	•	•
Block m	b_{1m}	b_{2m}		b_{nm}
Hash code	C_1	C_2		C_n

Figure 21.1 Simple Hash Function Using Bitwise XOR



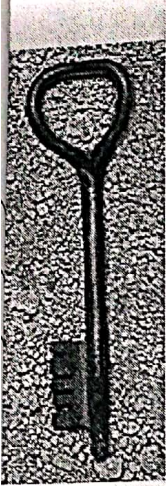
Keyed Hash Functions as MACs

- Want a MAC based on a hash function
 - because hash functions are generally faster
 - crypto hash function code is widely available
 - But hashing is internally has no key!
- Original proposal:

$$\text{KeyedHash} = \text{Hash}(\text{Key} \mid \text{Message})$$

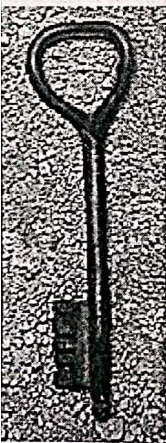
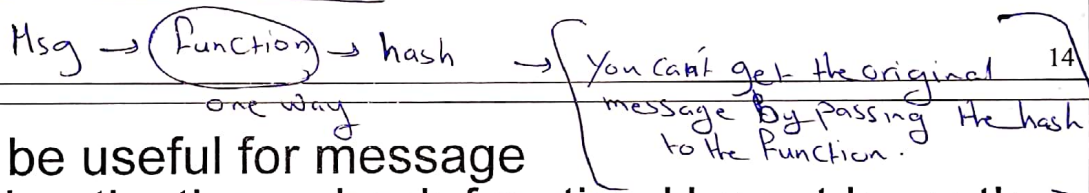
Concatenate

 - some weaknesses were found with this
- Eventually led to development of HMAC



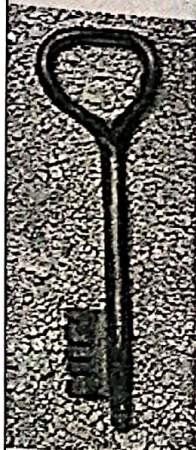
Security requirements

- **Pre-image:** if $h(m) = y$, m is a pre-image of y . \rightarrow harder than Collision
Can we find any message that has a hash value = y?
 - Each hash value typically has multiple pre-images.
 - **Collision:** a pair of (m, m') , $m \neq m'$, s.t. $h(m) = h(m')$. \rightarrow easier
Can we find any 2 messages $x \neq y$ such that $h(x) = h(y)$?
- A hash function is said to be:
- **Pre-image resistant** if it is computationally infeasible to find a pre-image of a hash value.
 - **Collision resistant** if it is computationally infeasible to find a collision.
Can we find any 2 messages
 - A hash function is a cryptographic hash function if it is collision resistant.



To be useful for message authentication, a hash function H must have the following properties:

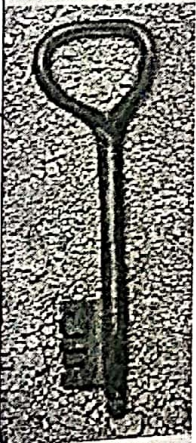
- Can be applied to a block of data of any size
- Produces a fixed-length output
- $H(x)$ is relatively easy to compute for any given x
- One-way or pre-image resistant
• Computationally infeasible to find x such that $H(x) = h$
- Computationally infeasible to find $y \neq x$ such that $H(y) = H(x)$
- Collision resistant or strong collision resistance
• Computationally infeasible to find any pair (x, y) such that $H(x) = H(y)$



* Birthday Problem

- Birthday problem: In a group of k people, what is the probability that at least two people have the same birthday?
 - Having the same birthday is a collision?
- Birthday paradox: $p \geq 1/2$ with k as small as 23.
- Consider a hash function $h: \{0,1\}^* \rightarrow \{0,1\}^n$.
- If we randomly generate k messages, the probability of having a collision depends on n .
- To resist birthday attack, we choose n to be sufficiently large that it will take an infeasibly large k to have a non-negligible probability of collision.

n bits hash
 n possible hash values
 → to find a pre-image on average I will inspect $\frac{2^n}{2} = 2^{n-1}$ messages
 → to find a collision → $\frac{n}{2}$

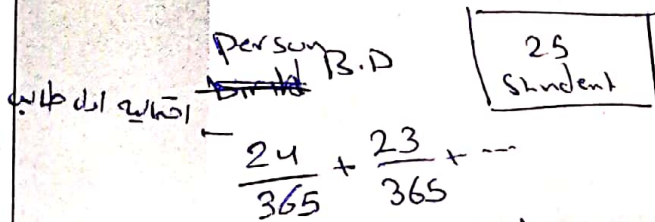


Collision-resistant hash functions

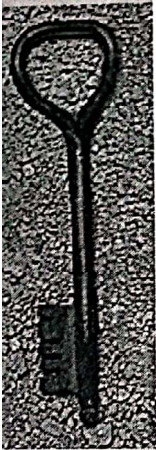
- ◆ Collision-resistant hash functions can be built from collision-resistant compression functions using **Merkle-Damgard construction.**

* class of 25 students
 What is the probability that any 2 have the same birthday?

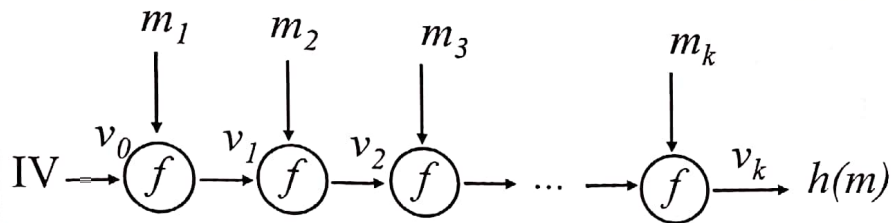
* pre-image: $\frac{25}{365}$



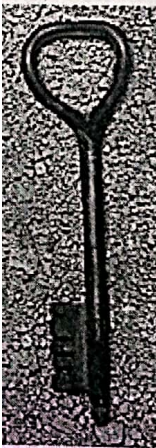
(ex) If the hash value is 200 bits, then to find collision we will try 2^{100} messages



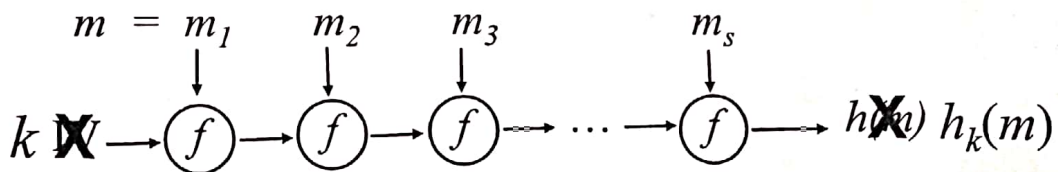
Merkle-Damgard Construction



Compression function $f : \{0,1\}^{n+b} \rightarrow \{0,1\}^n$



- Insecure: $MAC_k(m) = h(m)$ with $IV = k$.
(For simplicity, without padding)

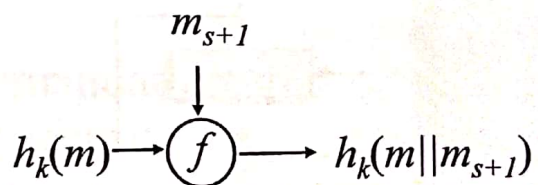


Replace initialization vector with Key.

- Easy to forge:

$(m', h_k(m'))$,

where $m' = m \parallel m_{s+1}$



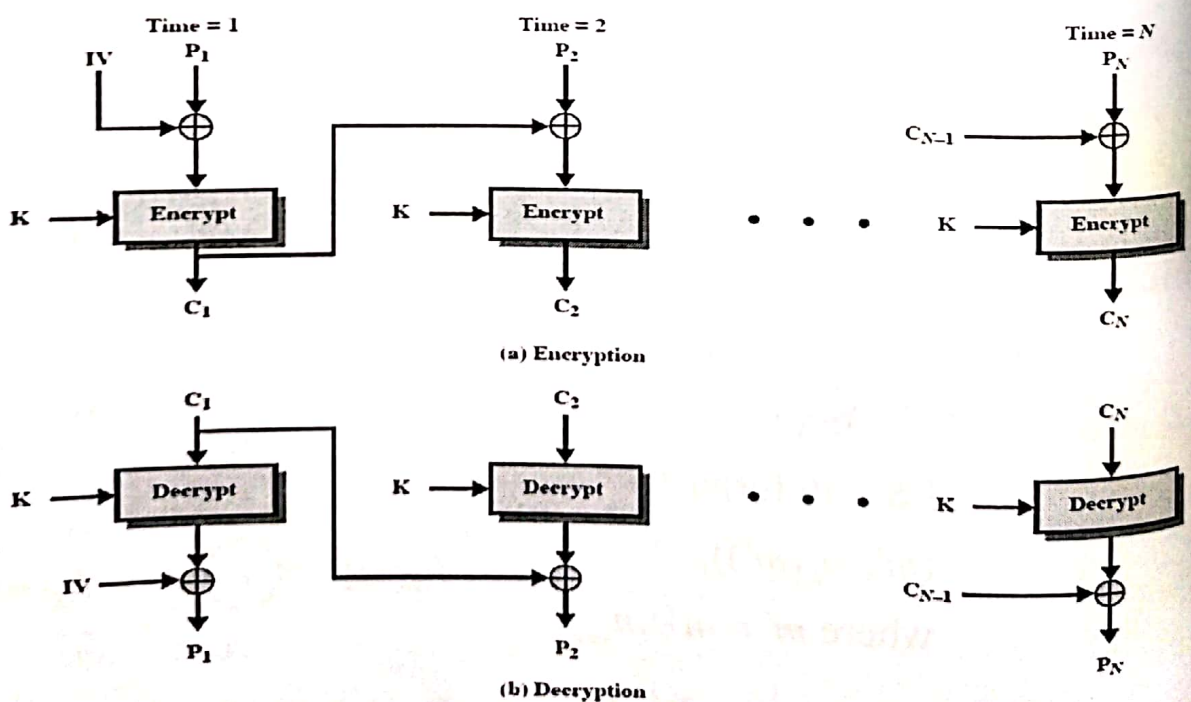
CMAC (Cipher-based MAC)

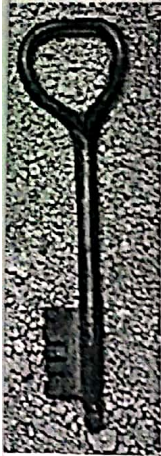
◆ “Hashless” MAC

- Uses an encryption algorithm (DES, AES, etc.) to generate MAC
- Based on same idea as cipher block chaining

◆ Compresses result to size of single block (unlike encryption)

CBC CMAC Overview





CMAC Facts

◆ Advantages:

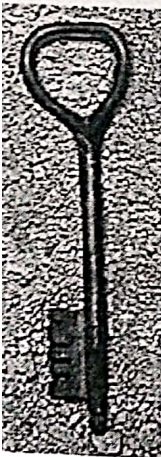
- Can use existing encryption functions
- Encryption functions have properties that resist preimage and collision attacks
- Most exhibit strong avalanche effect – minor change in message gives great change in resulting MAC

◆ Disadvantage:

- Encryption algorithms (particularly when chained) can be much slower than hash algorithms

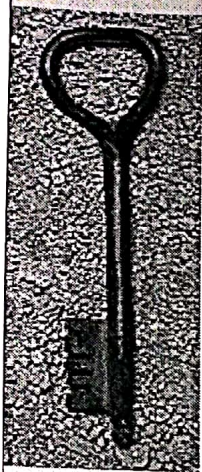
*

22



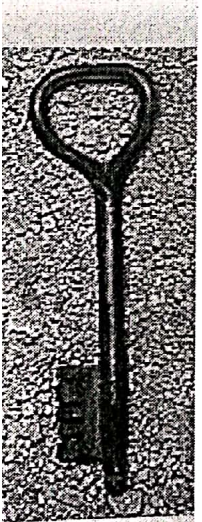
HMAC

- Interest in developing a MAC derived from a cryptographic hash code
 - Cryptographic hash functions generally execute faster
 - Library code is widely available
 - SHA-1 was not designed for use as a MAC because it does not rely on a secret key
- Issued as RFC2014
- Has been chosen as the mandatory-to-implement MAC for IP security
 - Used in other Internet protocols such as Transport Layer Security (TLS) and Secure Electronic Transaction (SET)

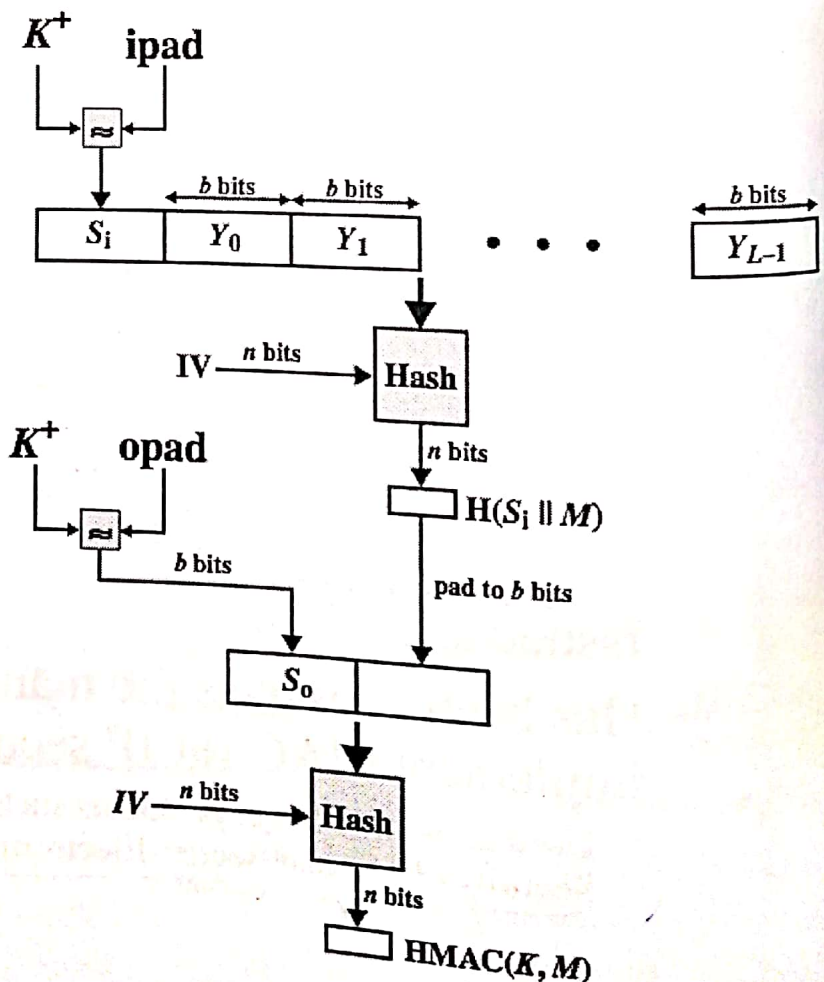


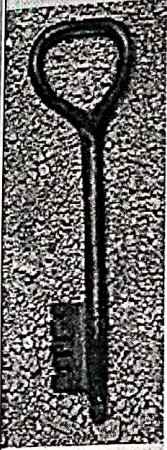
HMAC

- $HMAC(K, m) = H((K' \oplus \text{opad}) \parallel H((K' \oplus \text{ipad}) \parallel m))$, where
- H : is a cryptographic hash function, composed of multiple rounds with operations AND, OR, XOR, NOT, and SHIFT. Very efficient to compute.
 - K : is the secret key,
 - M : is the message to be authenticated,
 - K' : is another secret key, derived from the original key K (by padding K to the right with extra zeroes to the input block size of the hash function, or by hashing K if it is longer than that block size,
 - \parallel denotes concatenation,
 - opad is the outer padding (0x5c5c5c...5c5c, one-block long constant), and
 - ipad is the inner padding (0x363636...3636, one-block long constant).



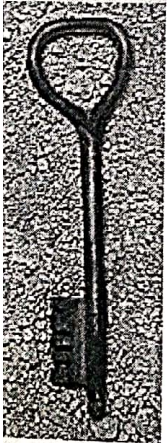
HMAC





Hash functions in practice

- ◆ MD5
 - 128-bit output
 - Introduced in 1991...collision attacks found in 2004...several extensions and improvements since then
 - Still widely deployed(!)
- ◆ SHA-1
 - 160-bit output
 - No collisions known, but theoretical attacks exist
- ◆ SHA-2
 - 256-/512-bit outputs



Secure Hash Algorithm (SHA)

- SHA was originally developed by NIST
- Published as FIPS 180 in 1993
- Was revised in 1995 as SHA-1
 - Produces 160-bit hash values
- NIST issued revised FIPS 180-2 in 2002
 - Adds 3 additional versions of SHA
 - SHA-256, SHA-384, SHA-512
 - With 256/384/512-bit hash values
 - Same basic structure as SHA-1 but greater security
- The most recent version is FIPS 180-4 which added two variants of SHA-512 with 224-bit and 256-bit hash sizes

Comparison of SHA Parameters

	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512	SHA-512/224	SHA-512/256
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$	$< 2^{128}$	$< 2^{128}$
Word size	32	32	32	64	64	64	64
Block size	512	512	512	1024	1024	1024	1024
Message digest size	160	224	256	384	512	224	256
Number of steps	80	64	64	80	80	80	80
Security	80	112	128	192	256	112	128

Hash Value (MAC) ←
Rounds ←

Notes:

1. All sizes are measured in bits.
2. Security refers to the fact that a birthday attack on a message digest of size n produces a collision with a work factor of approximately $2^{n/2}$.

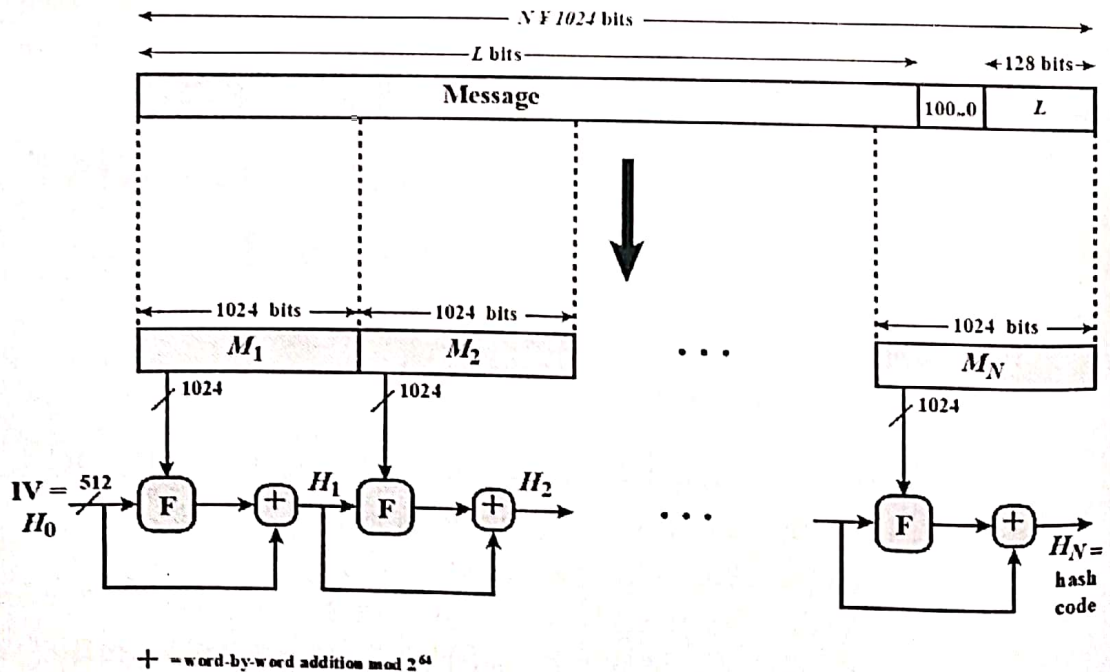


Figure 21.2 Message Digest Generation Using SHA-512

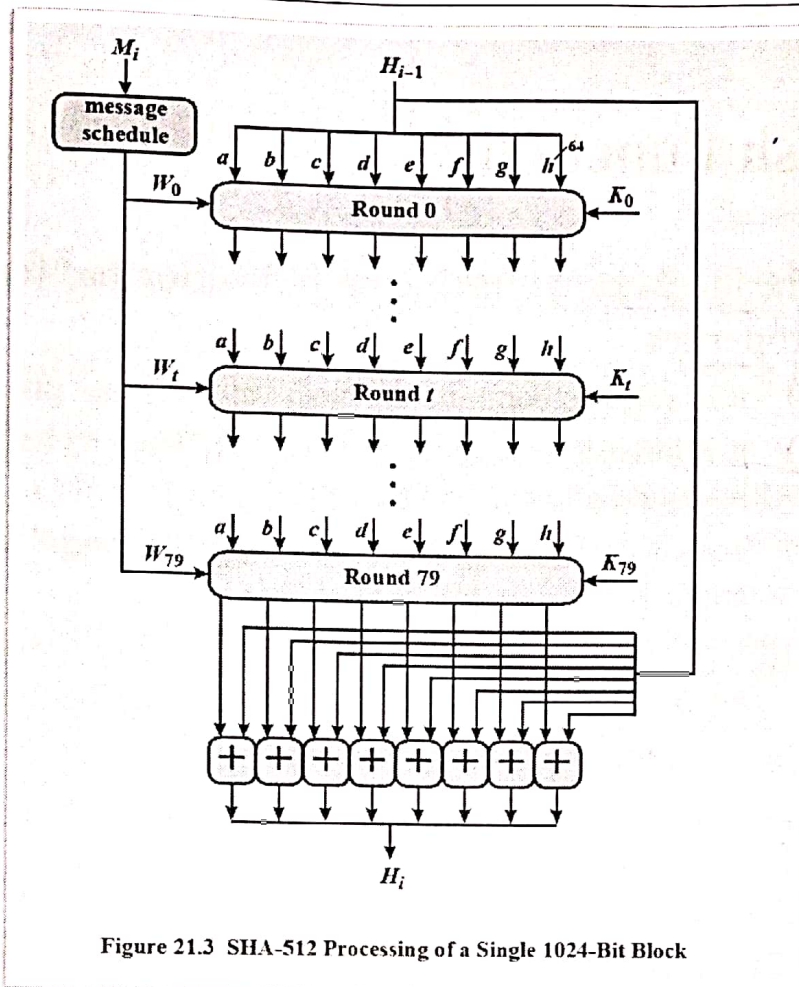


Figure 21.3 SHA-512 Processing of a Single 1024-Bit Block

SHA-3

- ◆ SHA-2 shares same structure and mathematical operations as its predecessors and causes concern
- ◆ Due to time required to replace SHA-2 should it become vulnerable, NIST announced in 2007 a competition to produce SHA-3

Requirements:

- Must support hash value lengths of 224, 256, 384, and 512 bits
- Algorithm must process small blocks at a time instead of requiring the entire message to be buffered in memory before processing it

Hash Function

- ◆ The ideal cryptographic hash function has four main properties:

- 1) it is quick to compute the hash value for any given message
- 2) it is infeasible to generate a message from its hash value except by trying all possible messages
- 3) a small change to a message should change the hash value so extensively
- 4) it is infeasible to find two different messages with the same hash value

one way
Function

غیر ممکن

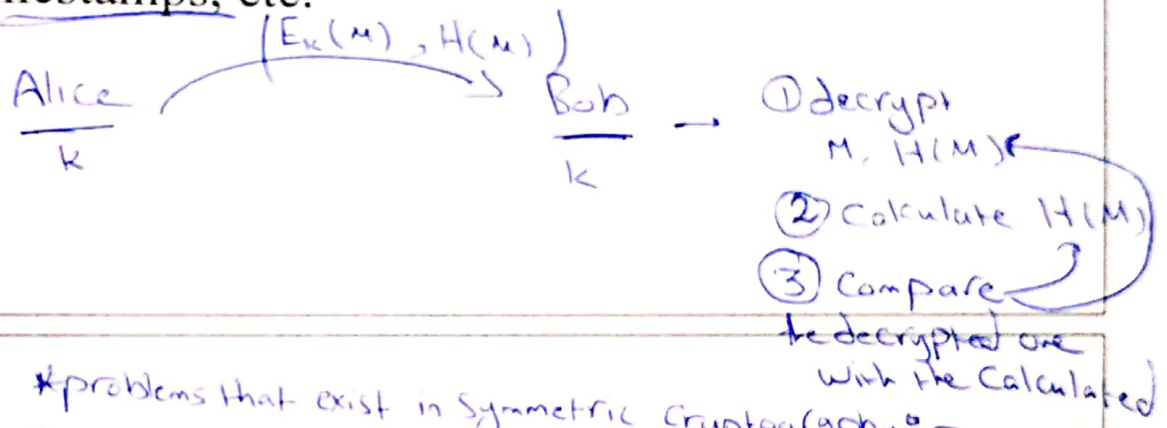
Encryption + integrity

- simultaneously protect confidentiality and authenticity of communications
 - often required but usually separate
- approaches
 - Hash-then-encrypt: $E_K(M \parallel H(M))$
 - MAC-then-encrypt: $E_{K_2}(M \parallel MAC_{K_1}(M))$
 - Encrypt-then-MAC: $(C=E_{K_2}(M), T=MAC_{K_1}(C))$
 - Encrypt-and-MAC: $(C=E_{K_2}(M), T=MAC_{K_1}(M))$
- decryption / verification straightforward
- but security vulnerabilities with all these



Replay attacks

- ◆ A MAC inherently cannot prevent replay attacks
- ◆ Replay attacks must be prevented at a higher level of the protocol!
 - (Note that whether a replay is ok is application-dependent.)
- ◆ Replay attacks can be prevented using nonces, timestamps, etc.



- * problems that exist in Symmetric Cryptography :-
- ① key agreement
 - ② 1-1 Message Authentication.



* Public Key Encryption

* Symmetric depends on:

- ① multiple rounds of substitutions & permutations

* Asymmetric depends on:

- ① Find large prime number (2048 bits) or (4096 bit)

Modified by: Dr. Ramzi Saifan

relatively easy but more complicated than (*) and (÷)

to check if N is prime? try to divide on 2, 3, ..., \sqrt{N}

↳ if didn't divide on any of them then it's prime

- multiply 2 digits: fast relatively.
- division 2 digits: fast relatively.

→ Factor a Composite: exponential time (n) — number of bits

→ given a large odd number, is it prime?

Prime Numbers

- Prime numbers only have divisors of 1 and itself
 - They cannot be written as a product of other numbers
- Any integer $a > 1$ can be factored in a unique way as

$$a = p_1^{a_1} * p_2^{a_2} * \dots * p_t^{a_t}$$

where $p_1 < p_2 < \dots < p_t$ are prime numbers and where each a_i is a positive integer

- This is known as the fundamental theorem of arithmetic

Table 8.1
Primes Under 2000

2	101	211	307	401	503	601	701	809	907	1009	1103	1201	1301	1409	1511	1601	1709	1801	1901
3	103	223	311	409	509	607	709	811	911	1013	1109	1213	1303	1423	1523	1607	1721	1811	1907
5	107	227	313	419	521	613	719	821	919	1019	1117	1217	1307	1427	1531	1609	1723	1823	1913
7	109	229	317	421	523	617	727	823	929	1021	1123	1223	1319	1429	1543	1613	1733	1831	1931
11	113	233	331	431	541	619	733	827	937	1031	1129	1229	1321	1433	1549	1619	1741	1847	1933
13	127	239	337	433	547	631	739	829	941	1033	1151	1231	1327	1439	1553	1621	1747	1861	1949
17	131	241	347	439	557	641	743	839	947	1039	1153	1237	1361	1447	1559	1627	1753	1867	1951
19	137	251	349	443	563	643	751	853	953	1049	1163	1249	1367	1451	1567	1637	1759	1871	1973
23	139	257	353	449	569	647	757	857	967	1051	1171	1259	1373	1453	1571	1657	1777	1873	1979
29	149	263	359	457	571	653	761	859	971	1061	1181	1277	1381	1459	1579	1663	1783	1877	1987
31	151	269	367	461	577	659	769	863	977	1063	1187	1279	1399	1471	1583	1667	1787	1879	1993
37	157	271	373	463	587	661	773	877	983	1069	1193	1283		1481	1597	1669	1789	1889	1997
41	163	277	379	467	593	673	787	881	991	1087		1289		1483		1693			1999
43	167	281	383	479	599	677	797	883	997	1091		1291		1487		1697			
47	173	283	389	487		683		887		1093		1297		1489		1699			
53	179	293	397	491		691				1097				1493					
59	181			499										1499					
61	191																		
67	193																		
71	197																		
73	199																		
79																			
83																			
89																			
97																			

to find large prime # :

① Probabilistic

② Deterministic

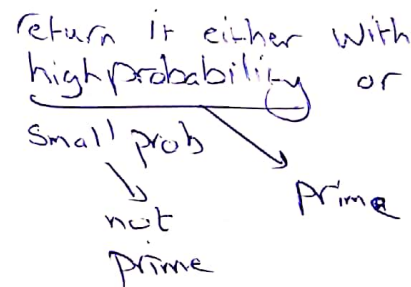
①

Miller-Rabin Algorithm (probabilistic)

Typically used to test a large number for primality

Algorithm is: TEST (n)

- Find integers k, q , with $k > 0, q$ odd, so that $(n - 1) = 2^k q$;
- Select a random integer $a, 1 < a < n - 1$;
- if $a^q \bmod n = 1$ then
 - return ("inconclusive");
- for $j = 0$ to $k - 1$ do
 - if $(a^{2^j q} \bmod n = n - 1)$ then
 - return ("inconclusive");
- return ("composite");



prob n is prime

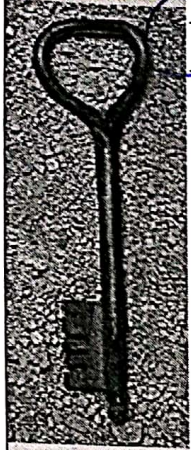
$$1 - \left(\frac{1}{4}\right)^x$$

Miller Rabin Usage

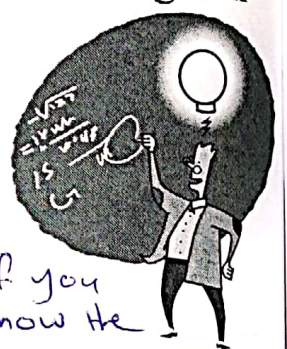
- ◆ It can be shown that given an odd number n that is not prime and a randomly chosen integer, a with $1 < a < n - 1$, the probability that TEST will return **inconclusive** (i.e., fail to detect that n is not prime) is less than $1/4$.
- ◆ Thus, if t different values of a are chosen, the probability that all of them will pass TEST (return inconclusive) for n is less than $(1/4)^t$. For example, for $t = 10$, the probability that a nonprime number will pass all ten tests is less than 10^{-6} .
- ◆ Thus, for a sufficiently large value of t , we can be confident that n is prime if Miller's test always returns **inconclusive**.
- ◆ invoke TEST (n) using randomly chosen values for a . If, at any point, TEST returns composite, then n is determined to be nonprime. If TEST continues to return **inconclusive** for t tests, then for a sufficiently large value of t , assume that n is prime.

②

Deterministic Primality Algorithm

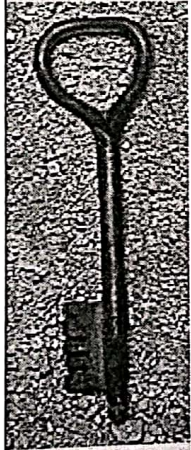


- Prior to 2002 there was no known method of efficiently proving the primality of very large numbers
- All of the algorithms in use produced a probabilistic result
- In 2002 Agrawal, Kayal, and Saxena developed an algorithm that efficiently determines whether a given large number is prime
 - Known as the AKS algorithm
 - Does not appear to be as efficient as Miller-Rabin algorithm



* PU & RR have Mathematical Relationship, even if you know the private key (PR), it's computationally infeasible to know the

(PR) ← public key: known to everybody
private key: only known to its owner



Public-Key Cryptography

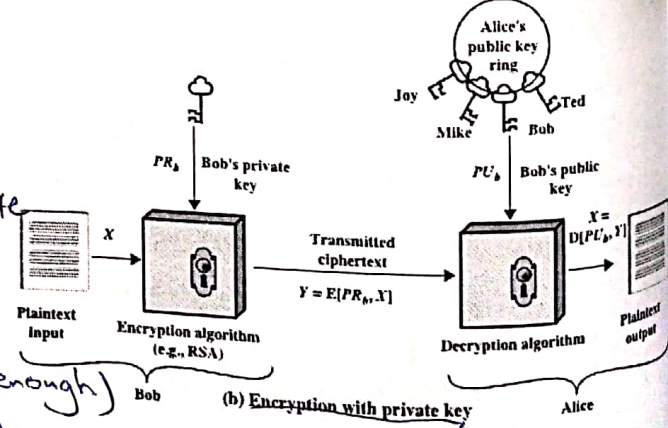
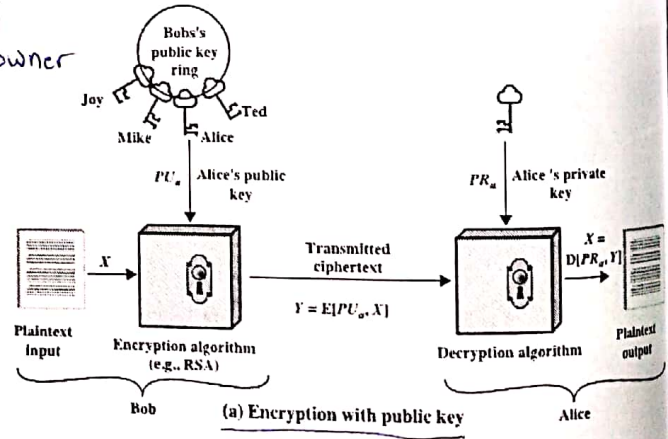


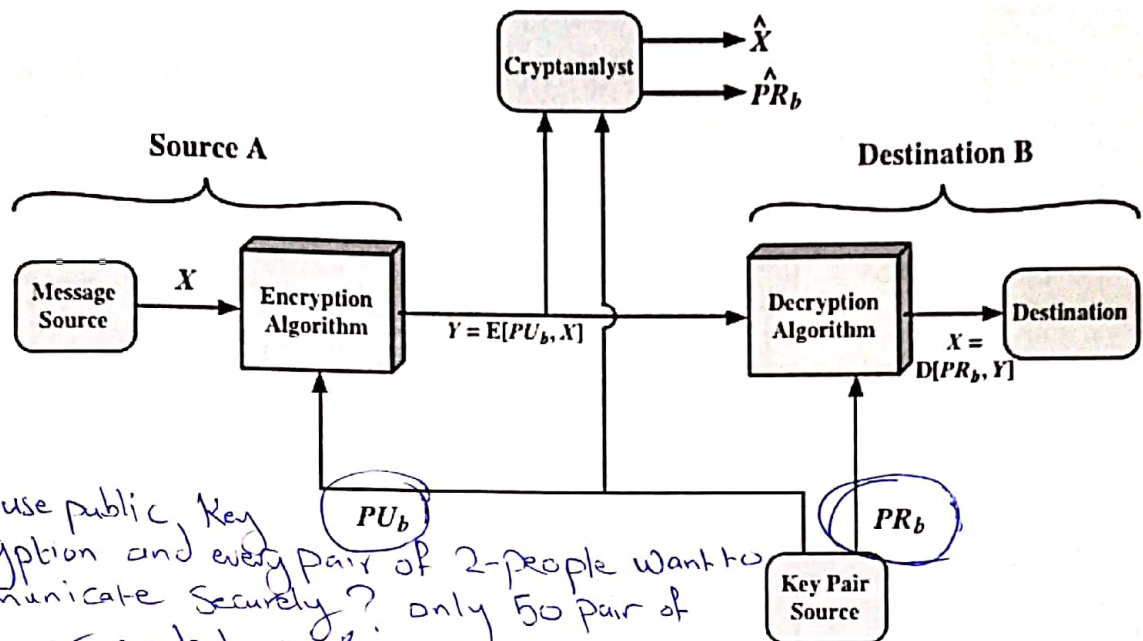
Figure 9.1 Public-Key Cryptography

ex: 50 person and everybody wants to communicate securely with everybody else

① $P_i, P_j \in [1, 50]$ send, only one of the fifty can read it (single key is enough)

② $P_i, P_j \in [1, 50]$ when P_i sends only P_j can read. How many keys needed?
~~49, 48, 47, ...~~ $49 + 48 + 47 + \dots + 1 = 49 \times 50 \approx 1200$ Keys

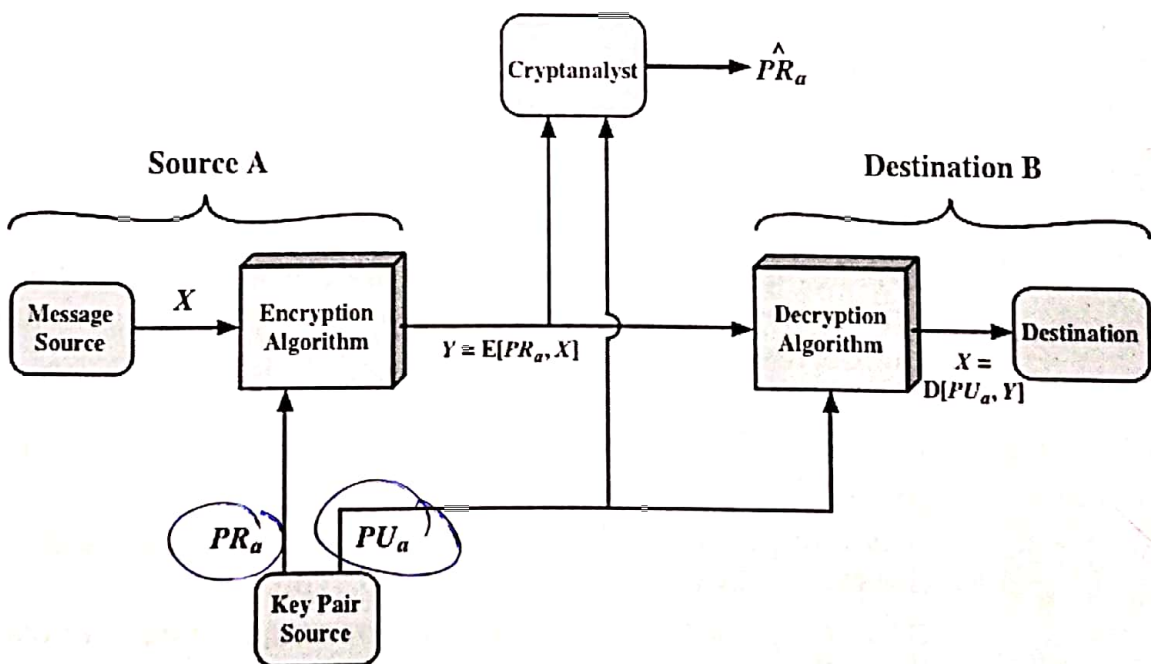
Public-Key Cryptosystem: Confidentiality



↳ 50 use public key encryption and every pair of 2-people want to communicate securely? only 50 pair of keys are needed pair/person.

↳ every person have pair of PU & PR → if we used PU key in encryption, we use PR key for decryption & vice-versa.

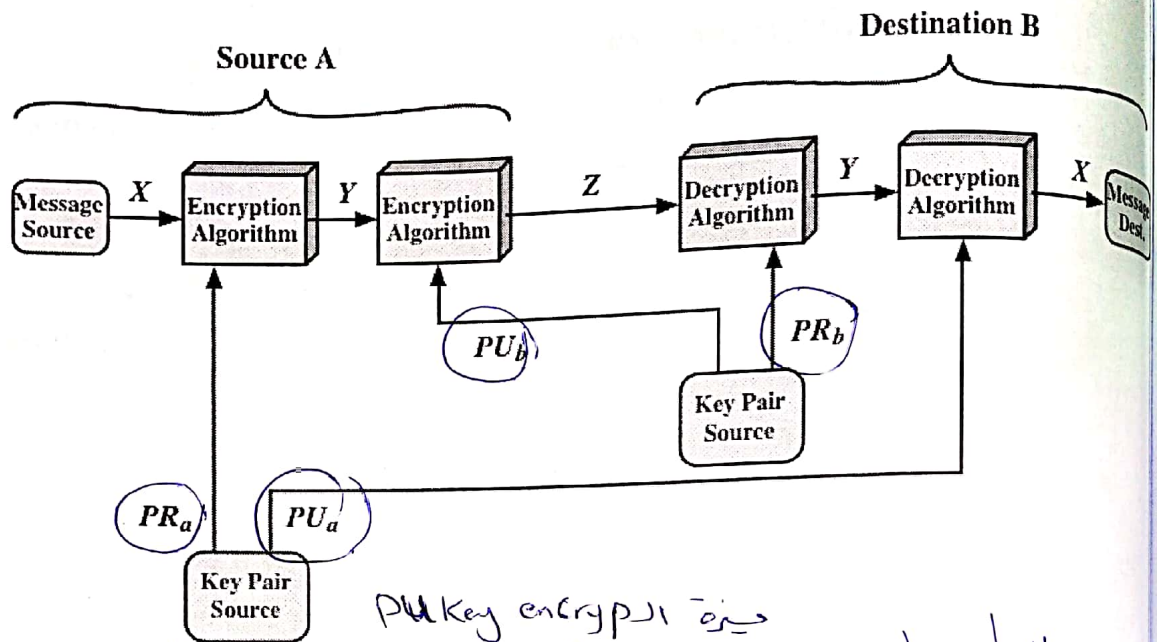
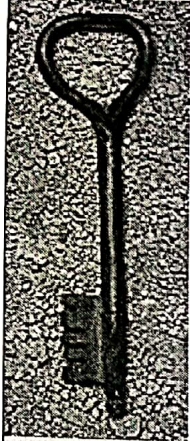
Public-Key Cryptosystem: Authentication



↳ don't encrypt a message using your own public key because you're the only one who can decrypt it using your private key (useless)

PU: Public PR: Private

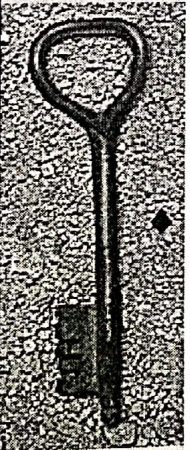
Public-Key Cryptosystem: Authentication and Confidentiality



PU & PR
~~PU & PR~~

PU key encryption

To derive one key from the other, some information are used and they are kept hidden afterwards. Without this information nobody can repeat the process.



Public-Key Requirements

easy but slower.

Conditions that these algorithms must fulfill:

- It is computationally easy for a party B to generate a pair (public-key PU_b , private key PR_b)
- It is computationally easy for a sender A, knowing the public key and the message to be encrypted, to generate the corresponding ciphertext
- It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message
- It is computationally infeasible for an adversary, knowing the public key, to determine the private key
- It is computationally infeasible for an adversary, knowing the public key and a ciphertext, to recover the original message
- The two keys can be applied in either order

Public-Key Requirements

- ◆ Need a trap-door one-way function → Like hash function.
 - A one-way function is one that maps a domain into a range such that every function value has a unique inverse, with the condition that the calculation of the function is easy, whereas the calculation of the inverse is infeasible
 - $Y = f(X)$ easy
 - $X = f^{-1}(Y)$ infeasible
- ◆ A trap-door one-way function is a family of invertible functions f_k , such that
 - $Y = f_k(X)$ easy, if k and X are known
 - $X = f_k^{-1}(Y)$ easy, if k and Y are known
 - $X = f_k^{-1}(Y)$ infeasible, if Y known but k not known
- ◆ A practical public-key scheme depends on a suitable trap-door one-way function

* p, q large prime, $n = p * q$, Find p or q given n
Comp infeasible (Factorization).

→ If you have p or q → $p = \frac{n}{q}$ or $q = \frac{n}{p}$

Rivest-Shamir-Adleman (RSA) Scheme

- ◆ Developed in 1977 at MIT by Ron Rivest, Adi Shamir & Len Adleman
- ◆ Most widely used general-purpose approach to public-key encryption
- ◆ Is a cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n
 - A typical size for n is 1024 bits, or 309 decimal digits

• n هو طول كل block و كل blocks الى مجموع الـ n

* $\phi(n)$: number of integers $< n$ that are relatively prime with n .

Table 8.2

Some Values of Euler's Totient Function $\phi(n)$

$\phi(p) = p - 1$ if p is prime
 $\phi(n) = (p-1)(q-1)$ while $n = p \times q$ and p, q primes.

n	$\phi(n)$
1	1
2	1
3	2
4	2
5	4
6	2
7	6
8	4
9	6
10	4

n	$\phi(n)$
11	10
12	4
13	12
14	6
15	8
16	8
17	16
18	6
19	18
20	8

n	$\phi(n)$
21	12
22	10
23	22
24	8
25	20
26	12
27	18
28	12
29	28
30	8

example

$$\phi(35) = (4) \times (6) = 24$$

$\begin{matrix} \uparrow & \uparrow \\ 5 \times 7 & (5-1) \quad (7-1) \end{matrix}$

RSA Algorithm (notes in the notebook).

- ◆ RSA makes use of an expression with exponentials
- ◆ Plaintext is encrypted in blocks with each block having a binary value less than some number n
- ◆ Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C

$$C = M^e \text{ mod } n$$

$$M = C^d \text{ mod } n = (M^e)^d \text{ mod } n = M^{ed} \text{ mod } n$$

- ◆ Both sender and receiver must know the value of n
- ◆ The sender knows the value of e , and only the receiver knows the value of d
- ◆ This is a public-key encryption algorithm with a public key of $PU = \{e, n\}$ and a private key of $PR = \{d, n\}$

Security :

No.

RSA Algorithm Slide.

$$M = \underbrace{01 \dots \dots}_{\text{block } M_1 < n}$$

$$C_1 = M_1^e \pmod n$$

everybody knows e, n (public)
only I know d (private)

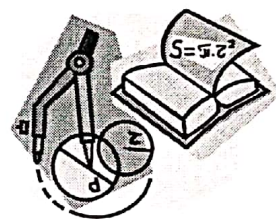
$$M_1 = C_1^d \pmod n$$

$$\begin{aligned} M_1 &= (M_1^e)^d \pmod n \\ &= M_1^{ed} \pmod n \end{aligned}$$

Algorithm Requirements

For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:

1. It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$
2. It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$
3. It is infeasible to determine d given e and n



Chosen randomly then apply Miller Rabin Algorithm.

Key Generation by Alice	
Select p, q	p and q both prime, $p \neq q$ (large prime numbers).
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\text{gcd}(\phi(n), e) = 1; 1 < e < \phi(n)$ (relatively prime)
Calculate d	$d = e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

IF $\phi(n) = 20$
 $e = 3$
 then $(d \times e) \bmod \phi(n) = 1$

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

$d = e^{-1} \bmod \phi(n)$
 $= 3^{-1} \bmod 20$
 $= 7$

Decryption by Alice with Alice's Private Key	
Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

* another example in notebook

Figure 9.5 The RSA Algorithm

* another example:

$$p = 11$$

$$q = 7$$

$$n = 77$$

$$\phi(n) = 60$$

$$e = 13$$

$$d = ??$$

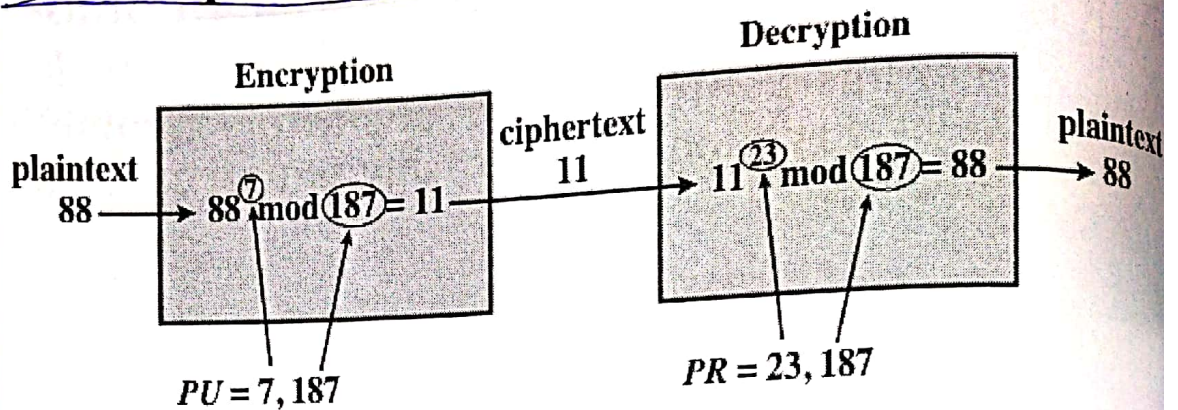
$$e \cdot d \pmod{\phi(n)} = 1$$

$$13 \cdot d \pmod{60} = 1$$

$$d = \boxed{}$$

d must be integer not fraction.

Example of RSA Algorithm



* Symmetric Enc :

- ① Message Auth
- ② Key agreement

Figure 9.6 Example of RSA Algorithm

→ Use public key.

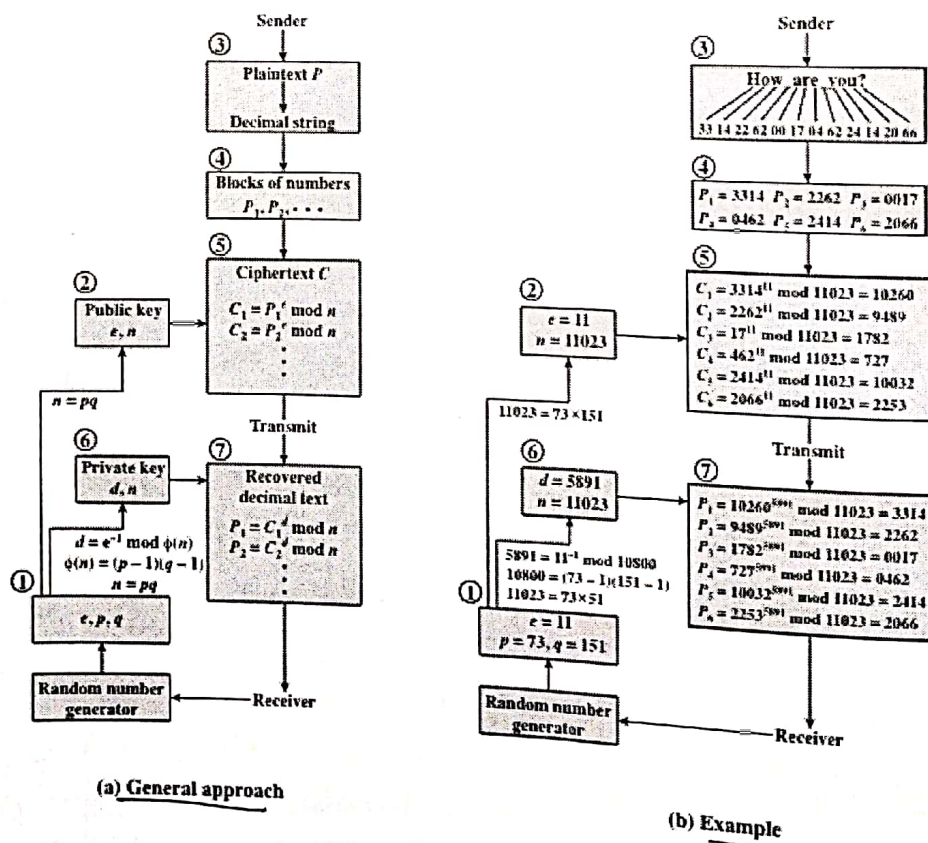


Figure 9.7 RSA Processing of Multiple Blocks



Fermat's Theorem

ex: If $p=11$ $a=9$

$$9^{10} = 1 \pmod{11}$$

- States the following:
 - If p is prime and a is a positive integer not divisible by p then

$$a^{p-1} = 1 \pmod{p}$$

- Sometimes referred to as Fermat's Little Theorem
- An alternate form is:
 - If p is prime and a is a positive integer then

$$a^p = a \pmod{p}$$

- Plays an important role in public-key cryptography



Euler's Theorem

- ◆ States that for every a and n that are relatively prime:

$$a^{\phi(n)} = 1 \pmod{n}$$

- ◆ An alternative form is:

$$a^{\phi(n)+1} = a \pmod{n}$$

Chinese Remainder Thm

- If p and q are prime, then for all x and a :
- $x = a \pmod{p}$ and $x = a \pmod{q}$ iff $x = a \pmod{pq}$

- Example:

- Suppose that $n = 2501 = 61 * 41$

- To calculate $V \pmod{2501}$:

- $V \pmod{61}$

- $V \pmod{41}$

ex: $p = 7$ $q = 11$ $n = 77$

$1053 \pmod{77}$

↳ $1053 \pmod{7}$

↳ $1053 \pmod{11}$

Correctness of RSA

- ◆ To show RSA is correct, we must show that encryption and decryption are inverse functions:

- $\text{En}(\text{De}(M)) = \text{De}(\text{En}(M)) = M = M^{\text{ed}} \pmod{n}$

- Since d and e are multiplicative inverses mod $\phi(n)$, there is a k such that:

→ • $ed = 1 + k * \phi(n), = 1 + k(p-1)(q-1)$

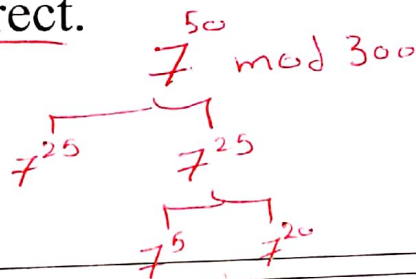
• $M^{\text{ed}} = M^{1+k(p-1)(q-1)} = M * (M^{p-1})^{k(q-1)}$

• By Fermat: $M^{p-1} = 1 \pmod{p}$

→ • $M^{\text{ed}} = M(1)^{k(q-1)} \pmod{p} = \underline{M \pmod{p}}$ #.

Correctness of RSA → slower than AES

- ◆ $M^{ed} = M(1)^{k(q-1)} \pmod p = M \pmod p$
- ◆ $M^{ed} = M(1)^{k(q-1)} \pmod q = M \pmod q$
- ◆ By Chinese Remainder Thm, we get:
- ◆ $M^{ed} = M \pmod p = M \pmod q = M \pmod{pq} = M \pmod n$
- ◆ Therefore, RSA reproduces the original message and is correct.



Exponentiation in Modular Arithmetic

- ◆ Both encryption and decryption in RSA involve raising an integer to an integer power, mod n
- ◆ Can make use of a property of modular arithmetic:
$$[(a \pmod n) \times (b \pmod n)] \pmod n = (a \times b) \pmod n$$
- ◆ With RSA you are dealing with potentially large exponents so efficiency of exponentiation is a consideration

Fast Exponentiation Algorithm

```

f = 1
for (i = k; i > 0; i--)
    f = (f * f) mod n;
    if (b_i == 1)
        f = (f * a) mod n;
return f;
    
```

Algorithm for computing $a^b \bmod n$, b is expressed as a binary $b_k b_{k-1} \dots b_0$

i	9	8	7	6	5	4	3	2	1	0
b_i	1	0	0	0	1	1	0	0	0	0
c	1	2	4	8	17	35	70	140	280	560
f	7	49	157	526	160	241	298	166	67	1

Result of the Fast Modular Exponentiation Algorithm for $a^b \bmod n$, where $a = 7$, $b = 560 = 1000110000$, and $n = 561$

If b is 4000 bits, how many squares? 4000

How many multiply by a ? based on # of 1's avg 2000.

Euclidean Algorithm

Ex: Find $\gcd(421, 111)$. use the Euclidean algorithm as follows:

$$421 = 111 \times 3 + 88$$

$$111 = 88 \times 1 + 23$$

$$88 = 23 \times 3 + 19$$

$$23 = 19 \times 1 + 4$$

$$19 = 4 \times 4 + 3$$

$$4 = 3 \times 1 + 1$$

$$3 = 1 \times 3 + 0$$

INPUT: Two non-negative integers a and b with $a \geq b$.

OUTPUT: $\gcd(a, b)$.

1. While $b > 0$, do
 1. Set $r = a \bmod b$,
 2. $a = b$,
 3. $b = r$
2. Return a .

The last non-zero remainder is 1 and therefore $\gcd(421, 111) = 1$.

Extended Euclidean Algorithm

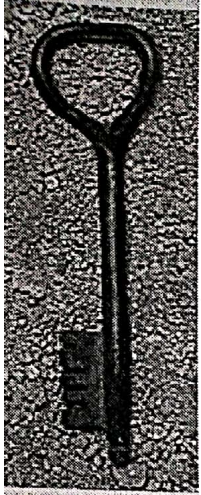
The following table can be used to calculate the the Euclidean algorithm and the Extended Euclidean algorithm

i	Quotient q_{i-1}	Remainder r_i	s_i	t_i
0	-	a	1	0
1	-	b	0	1
2	$\square \div \square = \square$	$\square - \square * \square = \square$	$\square - \square * \square = \square$	$\square - \square * \square = \square$
3	\square			
4				

Example

a=31 b= 12

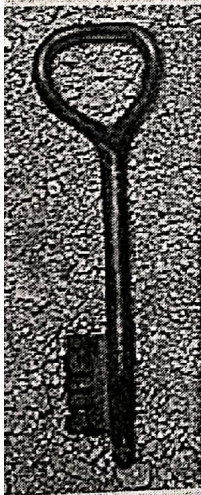
i	Quotient q_{i-1}	Remainder r_i	s_i	t_i
0	-	31	1	0
1	-	12	0	1
2	$31 \div 12 = 2$	$31 - 2 * 12 = 7$	$1 - 0 * 2 = 1$	$0 - 1 * 2 = -2$
3	$12 \div 7 = 1$	$12 - 1 * 7 = 5$	$0 - 1 * 1 = -1$	$1 - 1 * (-2) = 3$
4	$7 \div 5 = 1$	$7 - 1 * 5 = 2$	$1 - 1 * (-1) = 2$	$-2 - 1 * 3 = -5$
5	$5 \div 2 = 2$	$5 - 2 * 2 = 1$	$-1 - 2 * 2 = -5$	$3 - (-10) = 13$
	$2 \div 1 = 2$	$2 - 1 * 2 = 0$		



Efficient Operation Using the Public Key

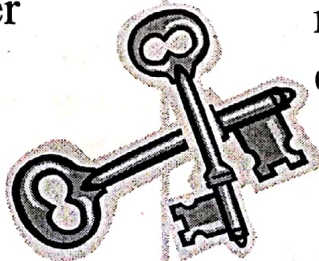
16 bits → 16 square
→ 2 multiply
bits

- ◆ To speed up the operation of the RSA algorithm using the public key, a specific choice of e is usually made
- ◆ The most common choice is 65537 ($2^{16} + 1$)
 - Two other popular choices are $e=3$ and $e=17$
 - Each of these choices has only two 1 bits, so the number of multiplications required to perform exponentiation is minimized
 - With a very small public key, such as $e = 3$, RSA becomes vulnerable to a simple attack



Key Generation

- ◆ Before the application of the public-key cryptosystem each participant must generate a pair of keys:
 - Determine two prime numbers p and q
 - Select either e or d and calculate the other
- ◆ Because the value of $n = pq$ will be known to any potential adversary, primes must be chosen from a sufficiently large set
 - The method used for finding large primes must be reasonably efficient



Public-Key Cryptanalysis

A public-key encryption scheme is vulnerable to a brute-force attack

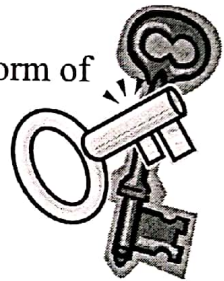
- Countermeasure: use large keys
- Key size must be small enough for practical encryption and decryption
- Key sizes that have been proposed result in encryption/decryption speeds that are too slow for general-purpose use
- Public-key encryption is currently confined to key management and signature applications

Another form of attack is to find some way to compute the private key given the public key

- To date it has not been mathematically proven that this form of attack is infeasible for a particular public-key algorithm

Finally, there is a probable-message attack

- This attack can be thwarted by appending some random bits to simple messages



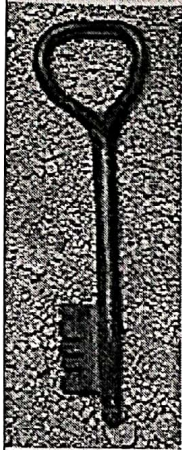
$C = M^e \pmod n$ → IF $M=56$ bit, try all possible M

Alice → c → Bob
Eve → $c^d \pmod n$, try possible d

Factoring Problem

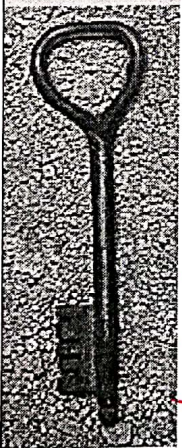
We can identify three approaches to attacking RSA mathematically:

- Factor n into its two prime factors. This enables calculation of $\phi(n) = (p-1) \times (q-1)$, which in turn enables determination of $d = e^{-1} \pmod{\phi(n)}$
- Determine $\phi(n)$ directly without first determining p and q . Again this enables determination of $d = e^{-1} \pmod{\phi(n)}$
- Determine d directly without first determining $\phi(n)$



Number of Decimal Digits	Number of Bits	Date Achieved
100	332	April 1991
110	365	April 1992
120	398	June 1993
129	428	April 1994
130	431	April 1996
140	465	February 1999
155	512	August 1999
160	530	April 2003
174	576	December 2003
200	663	May 2005
193	640	November 2005
232	768	December 2009

Table 9.5 Progress in RSA Factorization



MIPS-
Years
Needed
to
Factor

Million
Instruction
per
Second.

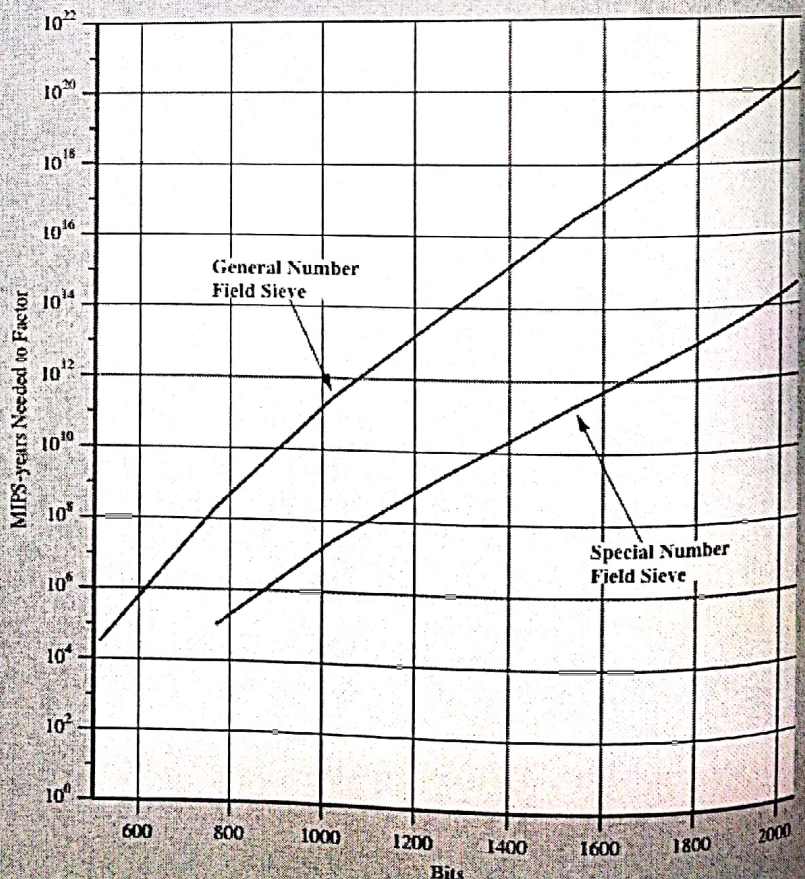
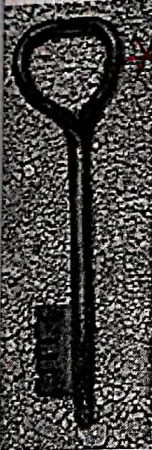


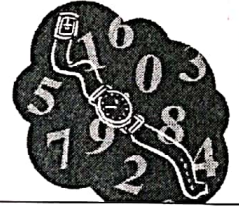
Figure 9.9 MIPS-years Needed to Factor



Timing Attacks

→ attack againsts all types of public key Encryption techniques

- ◆ Paul Kocher, a cryptographic consultant, demonstrated that a snooper can determine a private key by keeping track of how long a computer takes to decipher messages
- ◆ Are applicable not just to RSA but to other public-key cryptography systems
- ◆ Are alarming for two reasons:
 - It comes from a completely unexpected direction
 - It is a ciphertext-only attack



Countermeasures

Constant exponentiation time

- Ensure that all exponentiations take the same amount of time before returning a result; this is a simple fix but does degrade performance

Random delay

- Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack

Blinding

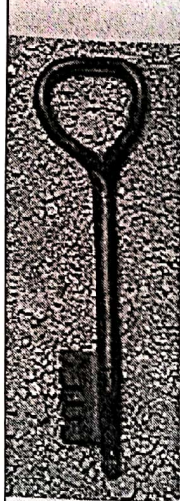
$$(a * c)^d$$

- Multiply the ciphertext by a random number before performing exponentiation; this process prevents the attacker from knowing what ciphertext bits are being processed inside the computer and therefore prevents the bit-by-bit analysis essential to the timing attack

التي
أضرب ال
C
Constant
ب
بعين ارفع
power
d

* In RSA you can generate the key while you're at home.

Misconceptions Concerning Public-Key Encryption

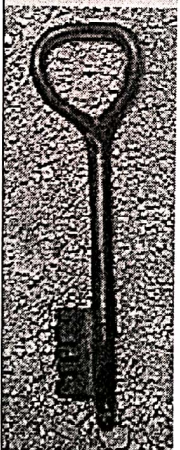


- ◆ Public-key encryption is more secure from cryptanalysis than symmetric encryption (AES)
- ◆ Public-key encryption is a general-purpose technique that has made symmetric encryption obsolete
- ◆ There is a feeling that key distribution is trivial when using public-key encryption, compared to the cumbersome handshaking involved with key distribution centers for symmetric encryption

* We still need to guarantee that a public key belongs to person.

Public Key (A) → person (A)

* public key is slower than AES.



Terminology Related to Asymmetric Encryption

Asymmetric Keys

Two related keys, a public key and a private key that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.

Public Key Certificate → has expiry date

A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key. The certificate indicates that the subscriber identified in the certificate has sole control and access to the corresponding private key.

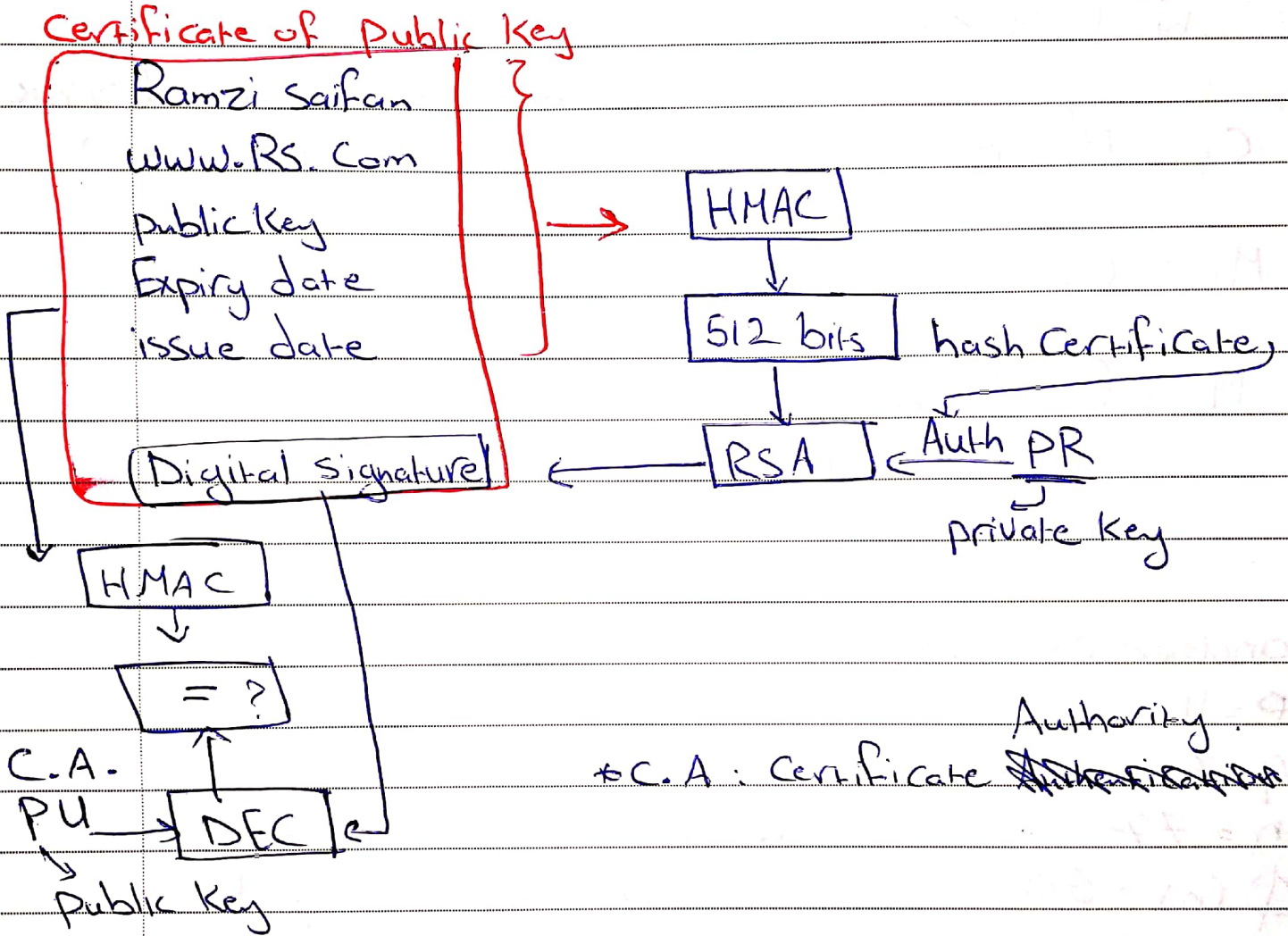
Public Key (Asymmetric) Cryptographic Algorithm

A cryptographic algorithm that uses two related keys, a public key and a private key. The two keys have the property that deriving the private key from the public key is computationally infeasible.

شرح على
الدفتر

Terminology Related to Asymmetric Encryption:

* Public Key Certificate:



Principles of Public-Key Cryptosystems

(شرح في دفتر)

- ◆ The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption:

Key distribution

- How to have secure communications in general without having to trust a KDC with your key

Digital Signatures

- How to verify that a message comes intact from the claimed sender

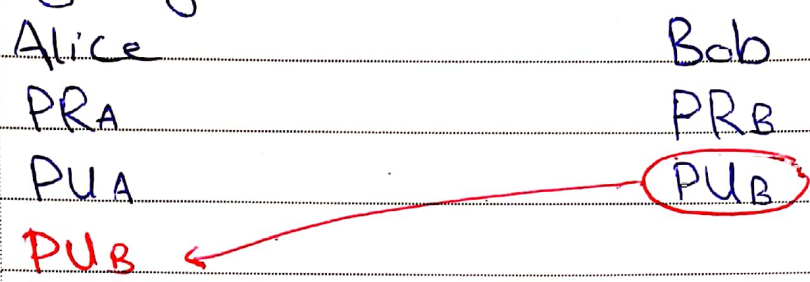
- ◆ Whitfield Diffie and Martin Hellman from Stanford University achieved a breakthrough in 1976 by coming up with a method that addressed both problems and was radically different from all previous approaches to cryptography

End

Questions

* Principles of public Key Crypto Systems

→ Key Agreement



Alice will generate an AES key randomly and Encrypt it using Bob's public key

$$C = E_{PU_B}^{Alice}(K_{AES}) \longrightarrow D_{PR_B}^{Bob}(C) = K_{AES}$$

Now they have An AES Key and they will use AES.
→ used RSA just to start.

Msg Auth :

① MAC :

- * HMAC
 - * CMAC
- both need Key

② Digital signature

$$M \mid E_{PR_A}(HMAC(M))$$

Concatenate

↓ Digital signature.

So; we used RSA & PU Key Enc in

- ① Key Agreement
- ② Digital signature.

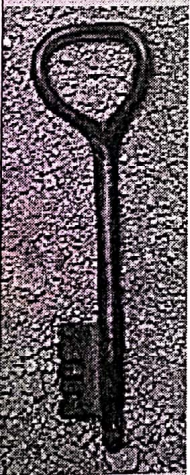
The Diffie-Hellman Algorithm



Key agreement- protocol

Modified by: Dr. Ramzi Saifan

Introduction



- ◆ Discovered by Whitfield Diffie and Martin Hellman
 - “New Directions in Cryptography”
- ◆ The point is to agree on a key that two parties can use for a symmetric encryption, in such a way that an eavesdropper cannot obtain the key.
- ◆ Diffie-Hellman key agreement protocol
 - Exponential key agreement
 - Allows two users to exchange a secret key
 - Requires no prior secrets
 - Real-time over an un-trusted network

Introduction

* D. H discrete log trap-door one way function.

- ◆ Based on the difficulty of computing discrete logarithms of large numbers.
- ◆ Requires two large numbers, one prime (P), and (G), a primitive root of P

* Any public key encryption is based on Trap-door one way function

$$x, k \xrightarrow{\text{easy}} f(x, k)$$

$f(x, k) \xrightarrow{\text{hard}}$ to find x if you don't have k

$f(x, k) \xrightarrow{\text{easy}}$ to find x given k .

HMAC (M, k) \rightarrow not trap-door, but one way function

$$P, g \xrightarrow{\text{easy}} n$$
$$n \xrightarrow{\text{hard}} P, g / n, P \xrightarrow{\text{easy}} g$$

Implementation

- ◆ p and g are both publicly available numbers
 - P is at least 512 bits

- ◆ Alice picks a private value "a" and send to Bob

- $A = g^a \text{ mod } p$

- ◆ Bob picks a private value "b" and sends to Alice:

- $B = g^b \text{ mod } p$

\rightarrow given A, g, p it's very hard to find a

* $g^i \text{ mod } p$ for all $i \in [1, p-1]$ results in all values in the range $1 - (p-1)$

* (شرح في دفتر)

* The Diffie-Hellman Algorithm Implementation :

Alice
Find a, g, p ^{using Miller-Rabin}
 $A = g^a \text{ mod } p$

Bob

Can choose a
value b
 $B = g^b \text{ mod } p$

send g, A, p →

← Send B to Alice

~~Bob doesn't know a~~
Alice doesn't know b

Bob doesn't know a

$$g^{a \times b} \text{ mod } p = B^a \text{ mod } p$$

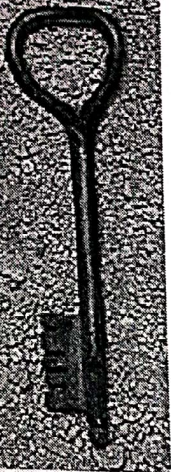
$$g^{a \times b} \text{ mod } p = A^b \text{ mod } p$$

$g^{a \times b} \text{ mod } p$ → Shared Value
Can be used as AES Key.

So,

- * private Key of Alice : a
 - * private Key of Bob : b
 - * public Key : A, B, G, p
- Key Agreement ✓
→ No Message Authentication guarantee X

Implementation



◆ Compute shared, private key:

– Alice received B and knows a , p and g , so she calculates:

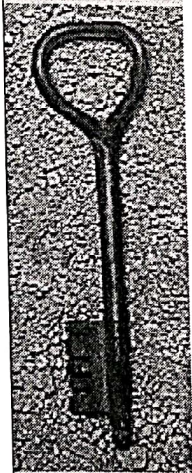
$$\bullet K_a = B^a \text{ mod } p$$

– Bob received A and knows b , p and g , so he calculates:

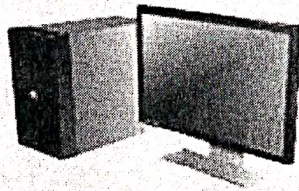
$$\bullet K_b = A^b \text{ mod } p$$

◆ Algebraically it can be shown that $K_a = K_b = K$

– Users now have a symmetric secret key to encrypt



Alice

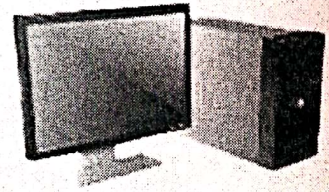


a, g, p

$$A = g^a \text{ mod } p$$

$$K = B^a \text{ mod } p$$

Bob



b

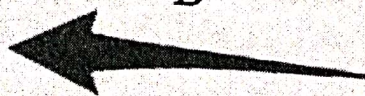
$$B = g^b \text{ mod } p$$

$$K = A^b \text{ mod } p$$

A, g, p



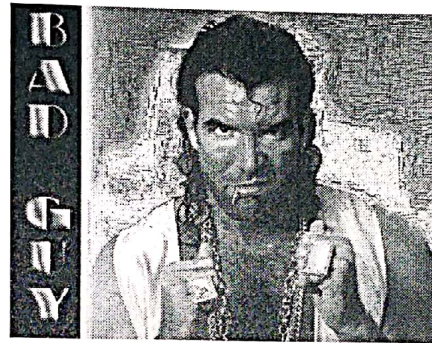
B



Example

- ◆ Bob and Alice are unable to talk on the untrusted network.

–Who knows who's listening?



Example

- ◆ Alice and Bob get public numbers
 - $P = 23$, $G = 9$
- ◆ Alice and Bob compute public values
 - $A = 9^4 \bmod 23 = 6561 \bmod 23 = 6$
 - – $B = 9^3 \bmod 23 = 729 \bmod 23 = 16$
- ◆ Alice and Bob exchange public numbers

Example

- ◆ Alice and Bob compute symmetric keys

$$k_a = B^a \bmod p = 16^4 \bmod 23 = \underline{9}$$

$$k_b = A^b \bmod p = 6^3 \bmod 23 = \underline{9}$$

- ◆ Alice and Bob now can talk securely!

Shared Key

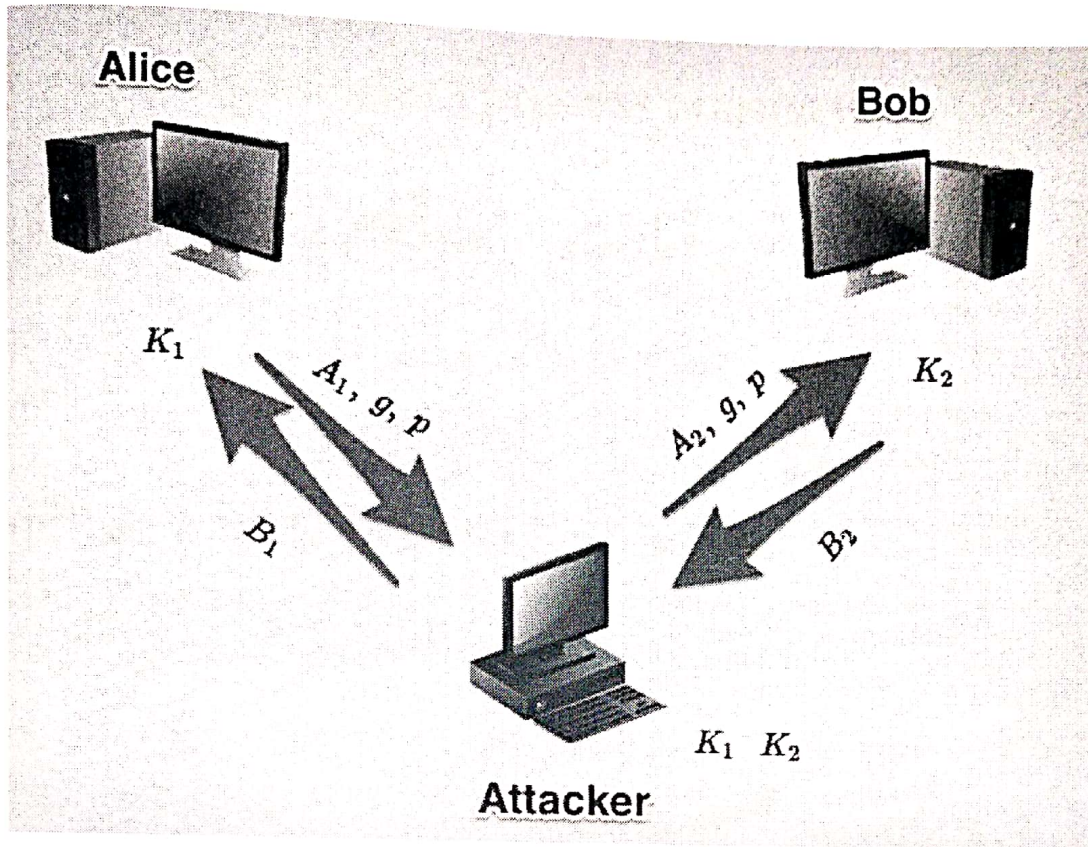
~~XXXXXXXXXX~~

Security of DH

- ◆ Suppose p is a prime of around 300 digits,
- ◆ and a and b at least 100 digits each.
- ◆ Discovering the shared secret given g , p , $g^a \bmod p$, and $g^b \bmod p$ would take longer than the lifetime of the universe, using the best known algorithm.
- ◆ This is called the discrete logarithm problem.

شرح في الدرس

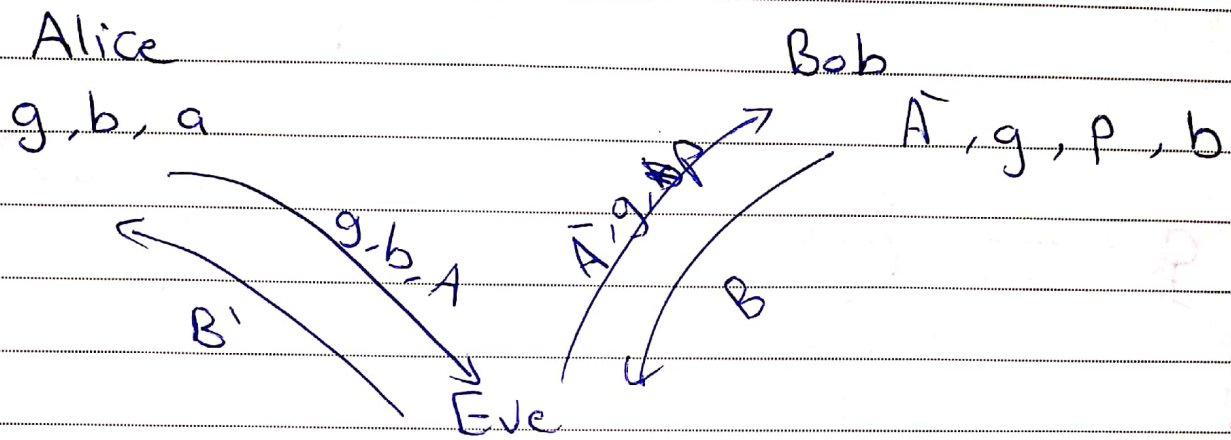
Man in the middle attack



Applications

- ◆ Diffie-Hellman is currently used in many protocols, namely:
 - Secure Sockets Layer (SSL)/Transport Layer Security (TLS)
 - Secure Shell (SSH)
 - Internet Protocol Security (IPSec)
 - Public Key Infrastructure (PKI)

* Implementation



$$g, p, A, a, B, b'$$

$$A' = g^{a'} \pmod{p}$$

$$B' = g^{b'} \pmod{p}$$

$$\text{Key} = B'^a \pmod{p}$$

$$\text{Key} = g^{ab'} \pmod{p}$$

$$\text{Key} = A'^b \pmod{p}$$

$$\text{Key} = g^{a'b} \pmod{p}$$

Key Alice \neq Key Bob
Eve Knows both Keys

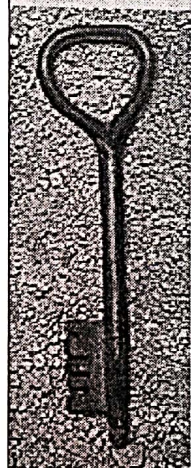
If Alice sent an ~~Encrypted~~ Encrypted to Bob, Bob can't read it because he doesn't know the Key, but Eve can read & Decrypt the Message

"Man in the Middle Attack"



User Authentication

Modified By: Dr. Ramzi Saifan



Authentication

- ◆ Verifying the identity of another entity
 - Computer authenticating to another computer
 - Person authenticating to a local/remote computer
- ◆ Important to be clear about what is being authenticated
 - The user?
 - The machine? A specific application on the machine?
 - The data?

handar

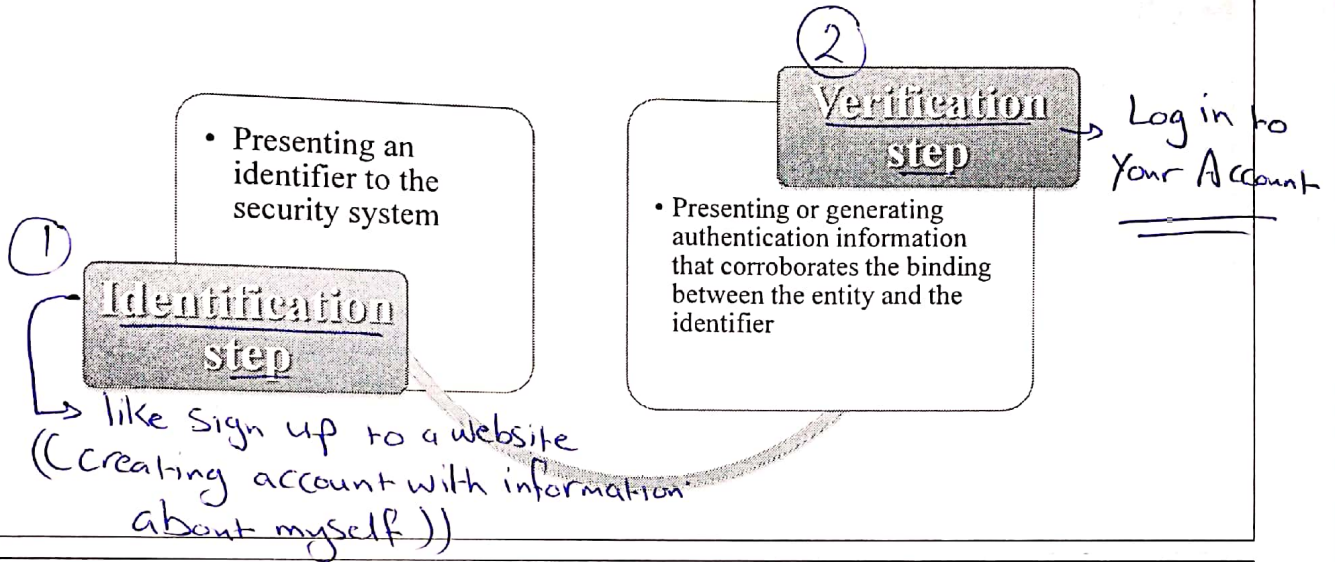
◆ Mutual authentication vs. unidirectional authentication

→ 2 persons talking to each other for the first time

like login to Facebook.

Remote User-Authentication Principles

- ◆ An authentication process consists of two steps:



Authentication

- ◆ Authentication may be based on ^{most used}

1. What you know like passwords, PIN code
2. What you have → مثل الهوية
3. What you are → مثل البصمة
4. What you do → Your typing behaviour (how fast, time you keep pressing the letter, time you take to press the letter ---)

- Examples? Tradeoffs?

- Others?

- * What you know : efficient, fast, easy, no cost // Can be stolen, weak password, if lost it will cause problems
- * What you have : not easy to steal // if lost it will cause problems
- * What you are : No one can use it instead of me // slower & not convenient

IP / MAC addresses . Address-based authentication

- ◆ Is sometimes used
- ◆ Generally not very secure → IP بقدر اعل Fake .
 - Relatively easy to forge source addresses of network packets تزوير
- ◆ But can be useful if the adversary does not know what IP address to forge
 - E.g., IP address of a user's home computer

لا سيج VPN في البقر .

Multi-factor Authentication

password something else → like phone number .

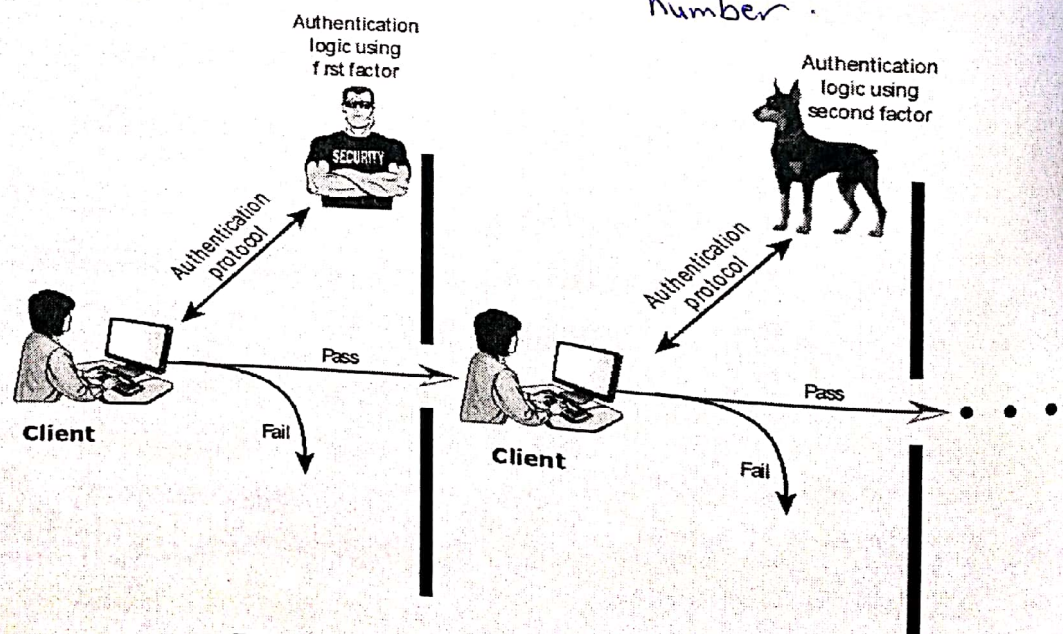
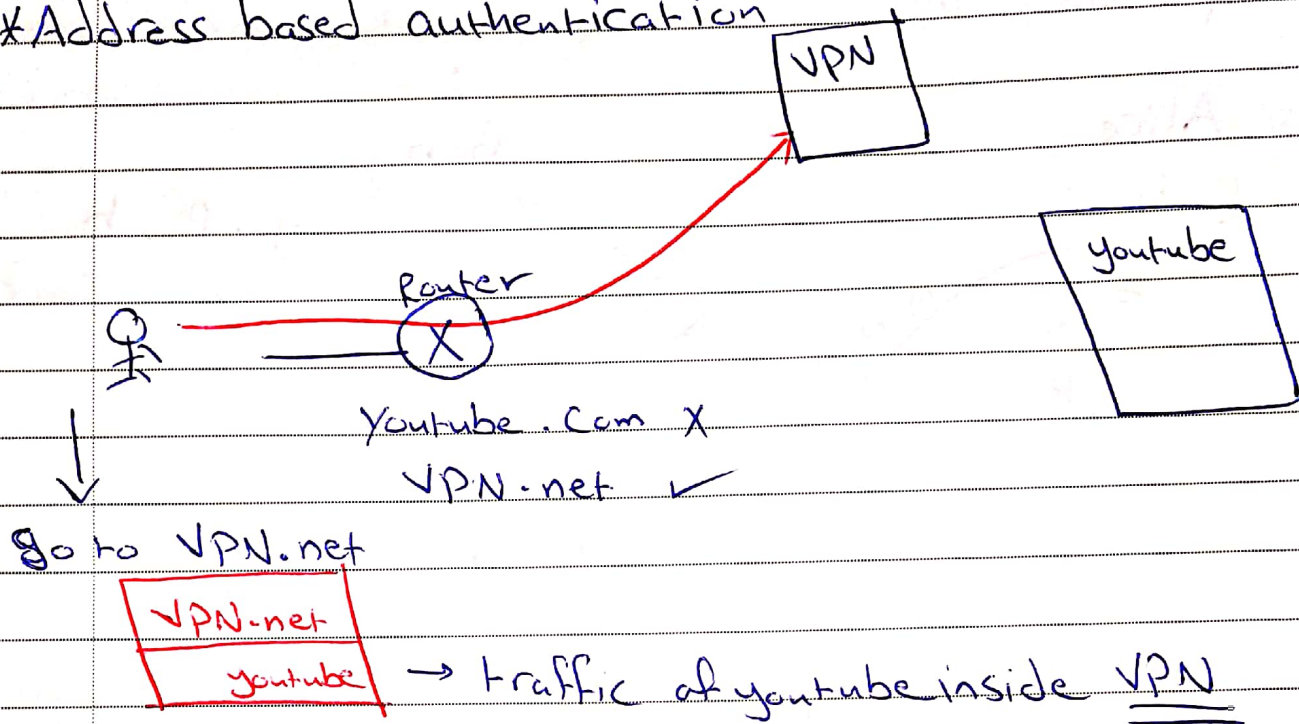


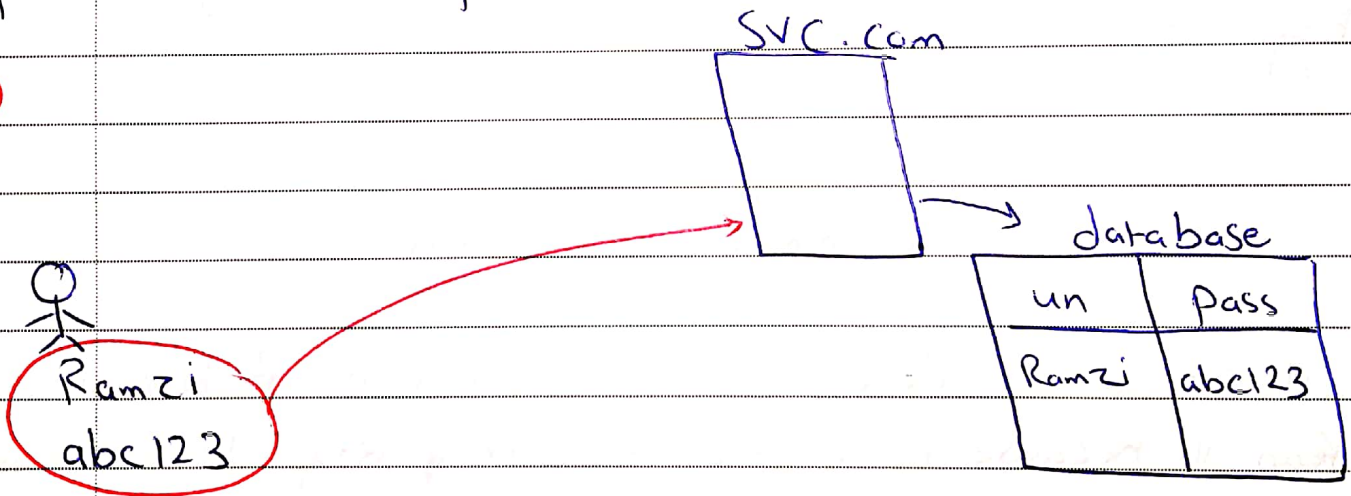
Figure 3.2 Multifactor Authentication

*Address based authentication



*password-based protocols :-

①



* password is sent as clear text ! No Encryption
 * IF someone Hacked database will know passwords.

Password-based protocols

◆ Basic idea

لا شرح في دفتر

- User has a secret password
- System checks password to authenticate user

◆ Issues

- How is password stored?
- How does system check password?
- How easy is it to guess a password?
 - Difficult to keep password file secret, so best if it is hard to guess password even if you have the password file

◆ Distinguish on-line attacks vs. off-line attacks

Basic password scheme

User

Password file

kiwifruit

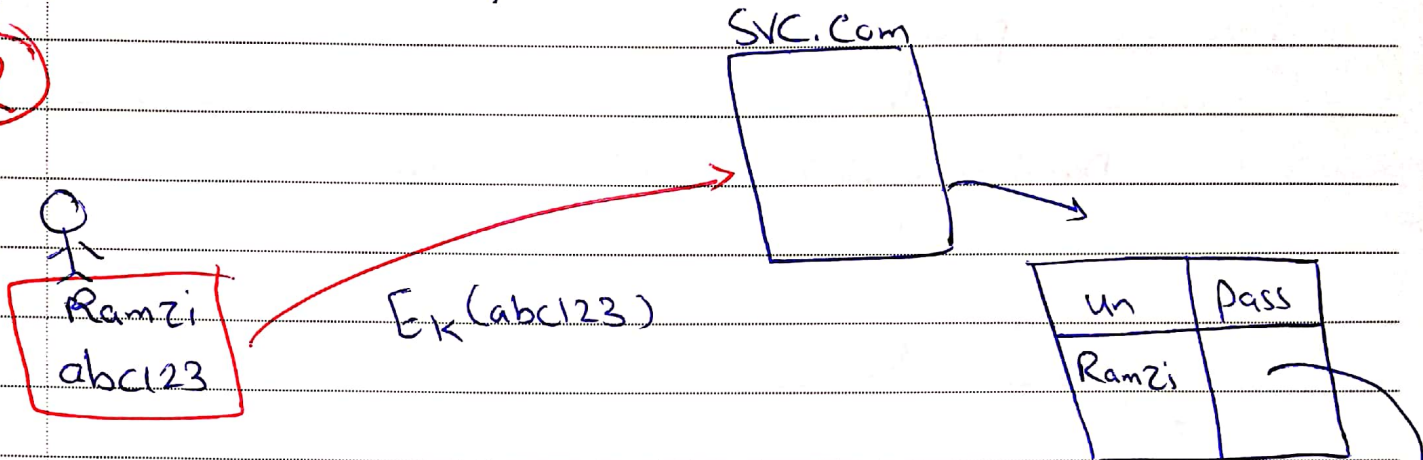
exrygbzyf
kgnosfix
ggjoklbsz
...
...

hash function



* password-based protocols :-

②



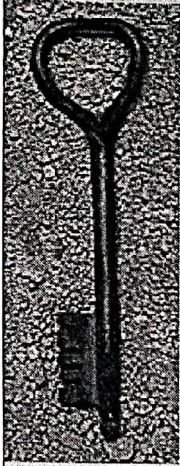
* password not saved
Encrypted !!

* passwords in data base are saved Hashed
because Hash is one way function.

HMAC(abc123)

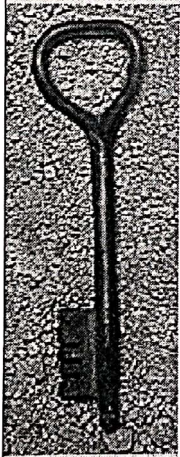
⇒ Approach to avoid this ?

login $\frac{\text{Hash}}{\text{username}}$ / $\frac{\text{username}}{\text{Hash}}$



Basic password scheme

- ◆ Hash function $h : \text{strings} \rightarrow \text{strings}$ ^{variable size} \rightarrow fixed size
 - Given $h(\text{password})$, hard to find password
 - No known algorithm better than trial and error
 \hookrightarrow brute force attack
- ◆ User password stored as $h(\text{password})$
- ◆ When user enters password
 - System computes $h(\text{password})$
 - Compares with entry in password file
- ◆ No passwords stored on disk
 - * No passwords stored on database



Unix password system

- ◆ In past UNIX systems, password used modified DES (encryption algorithm) as if it were a hash function

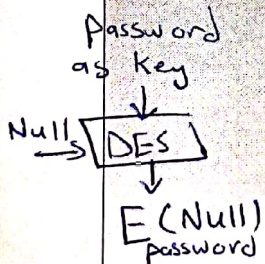
- Encrypts NULL string using password as the key (truncates passwords to 8 characters!)

- Caused artificial slowdown: ran DES 25 times

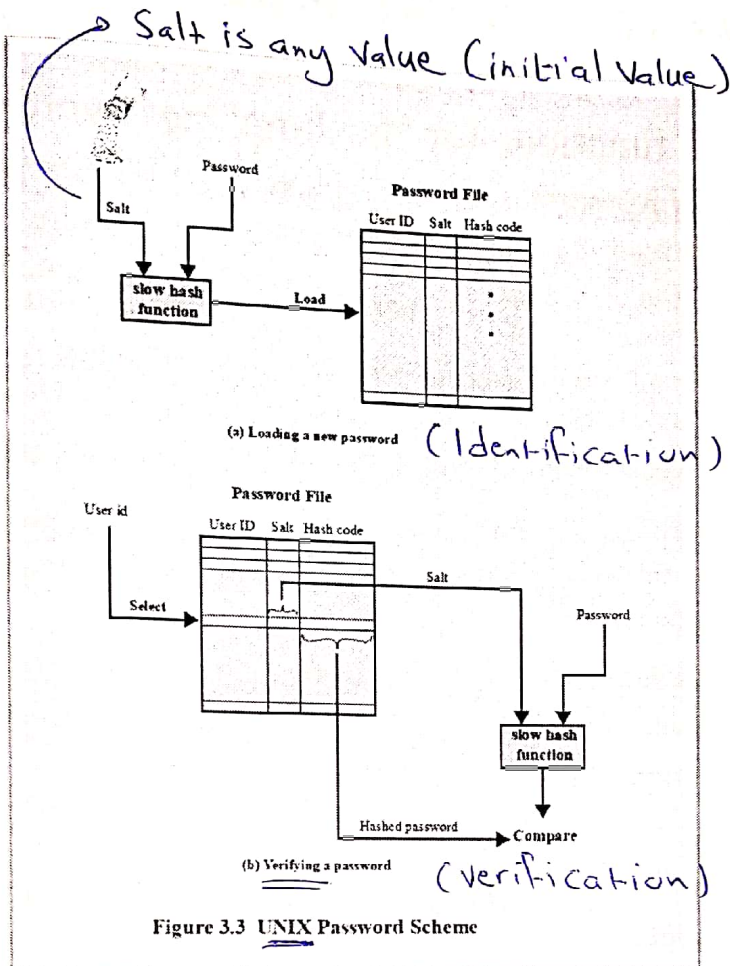
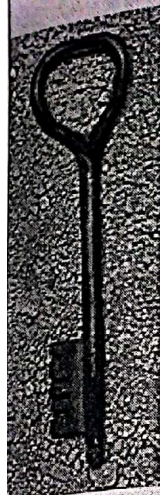
- ◆ Also stored password file in directory: /etc/passwd/

- World-readable (anyone who accessed the machine would be able to copy the password file to crack at their leisure)

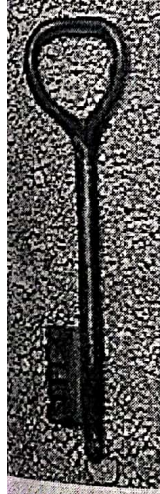
- Contained userIDs/groupIDs used by many system programs
- Can instruct modern UNIXes to use MD5 hash function



offline attack



Improved Implementations

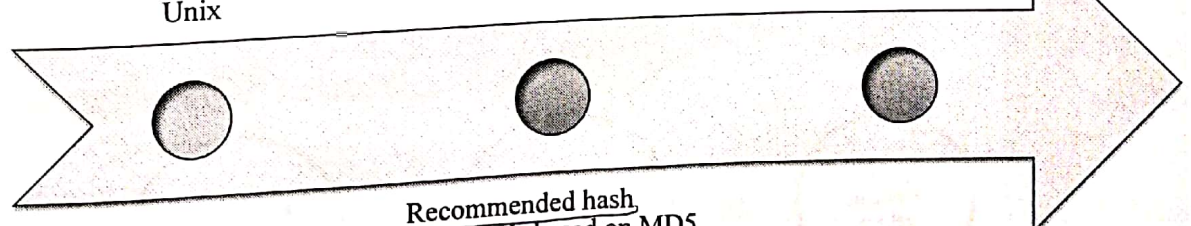


Much stronger hash/salt schemes available for Unix

Most secure UNIX password system.

OpenBSD uses Blowfish block cipher based hash algorithm called Bcrypt

- Most secure version of Unix hash/salt scheme
- Uses 128-bit salt to create 192-bit hash value

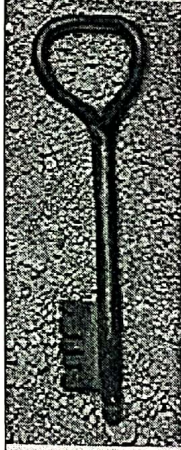


Recommended hash function is based on MD5

- Salt of up to 48-bits
- Password length is unlimited
- Produces 128-bit hash
- Uses an inner loop with 1000 iterations to achieve slowdown

* All the passwords in windows (except in domain Controller Configuration) are stored in configuration database (SAM)

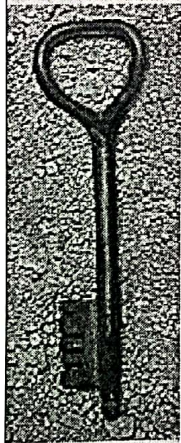
Windows NT/2k/XP/Vista Password



الدفتر
الدفتر

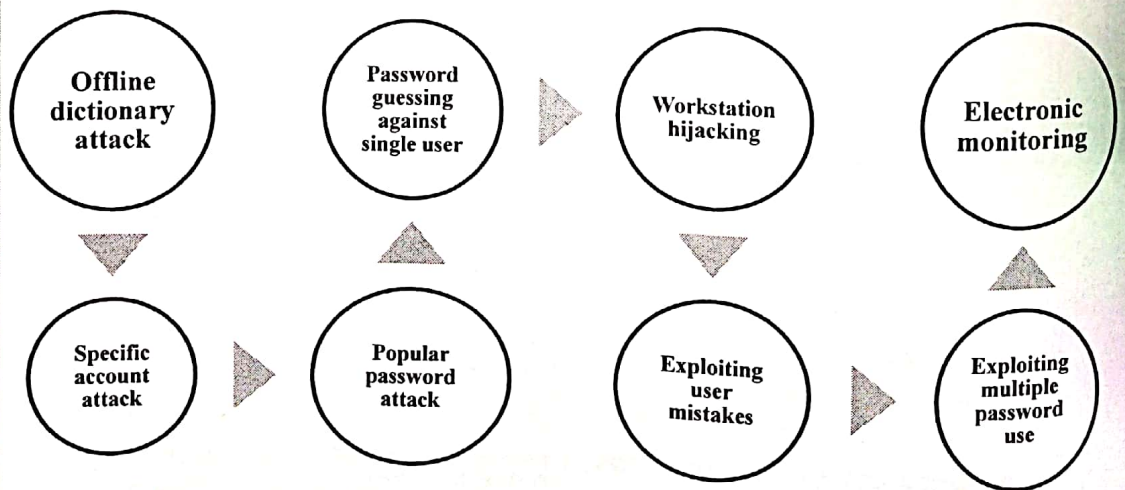
- ◆ Uses 2 functions for "hashing" passwords:
 1. LAN Manager hash (LM hash)
 - Password is padded with zeros until there are 14 characters.
 - It is then converted to uppercase and split into two 7-character pieces
 - Each half is encrypted using an 8-byte DES (data encryption standard) key to encrypt the constant "KGS!@#\$\$%&"
 - Result is combined into a 16-byte, one way hash value
 2. NT hash (NT hash)
 - Converts password to Unicode and uses MD4 hash algorithm to obtain a 16-byte value
- ◆ Hashes stored in Security Accounts Manager (SAM)
 - Locked within system kernel when system is running.
 - Location - C:\WINNT\SYSTEM32\CONFIG
- ◆ SYSKEY
 - Utility which moves the encryption key for the SAM database off of the computer

* to steal the password: there are a few different options here depending on the level of access you have to the machine: locally or remotely.



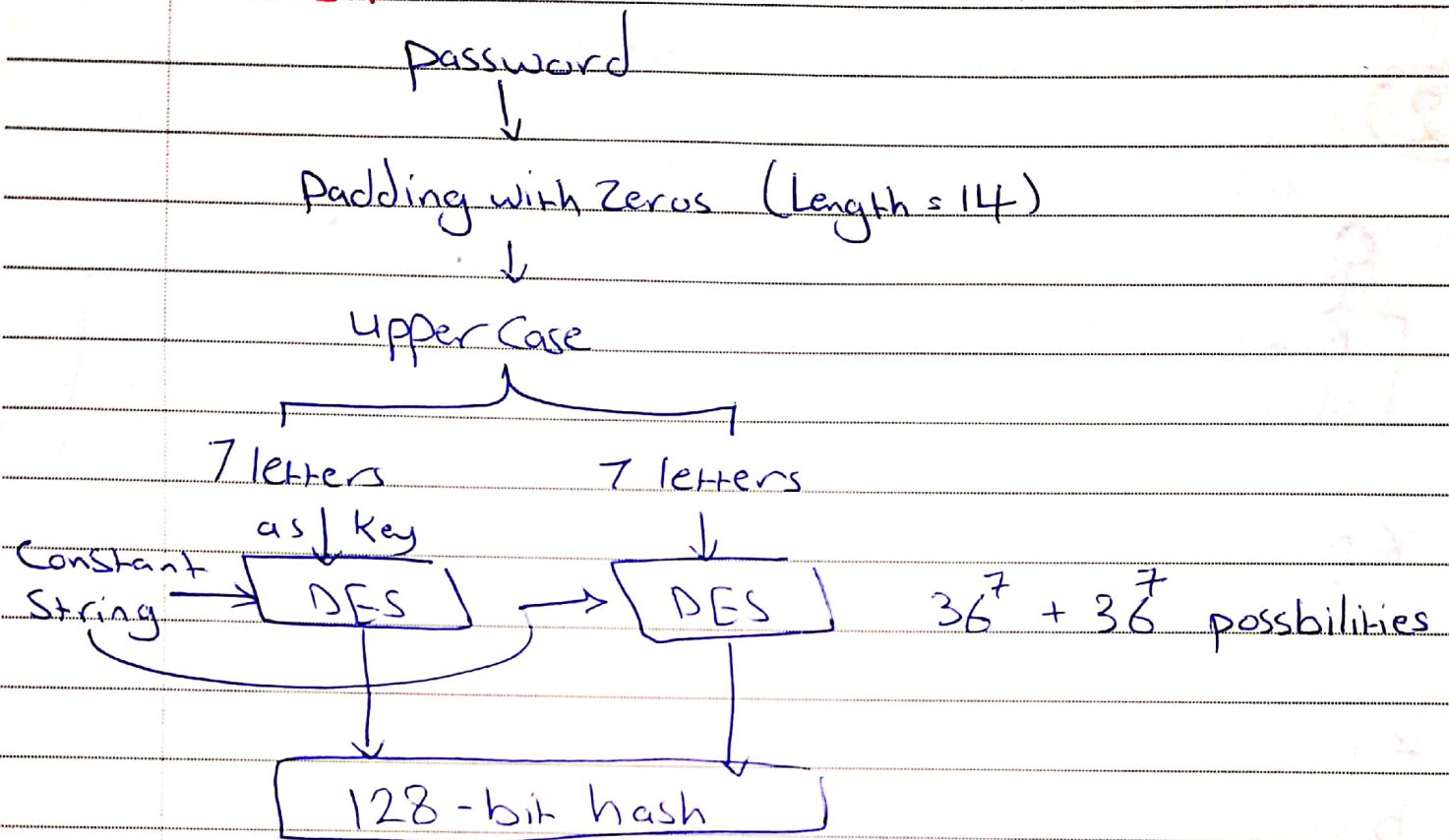
Password Vulnerabilities

3 missing slides here
(مكتوبين على الدفتر)



Windows NT/2k/XP/Vista/Password

* LM Hash :-



* اذا كان password من 14 حرفين او اكثر
 password

* 36 حرفين من الاعداد و الحروف
 !! upper password

Stealing of the Hash - Remote

- * In this case, passwords are dumped from the memory of remote system, by loading the password dumping program from remote.
- * This requires at least an administrative account.
- * This can be done using tools such as:
 - pwdump: <http://www.fooofus.net/~fizzgig/pwdump/>
 - Fgdump: <http://www.fooofus.net/goons/fizzgig/fgdump/>
 - Ophcrack: <http://ophcrack.sourceforge.net/>

Stealing the Hash - Local

- * Locally: Here you need physical access to the machine.
- * At this point there are two cases: Running System & offline system.
 - **Running System:** In this case, a local administrator account is required to download hashes from the memory.
 - pwdump: <http://www.fooofus.net/~fizzgig/pwdump/>
 - Fgdump: <http://www.fooofus.net/goons/fizzgig/fgdump/default>
 - SAMinside: <http://insidepro.com/eng/saminside.shtml> htm
 - Ophcrack: <http://ophcrack.sourceforge.net>
 - l0phtCrack: <http://www.l0phtcrack.com/>
- * All these tools are very simple to use you just have to run them and wait for the hashes. Also some of these allow you to directly crack hashes.

Offline Local Hash Steal

* If you have physical access to the off-line machine, you have a few more options than if you had a live system.

* You can:

- still steal hashes (using previous tools)
- overwrite hashes or
- bypass Windows Login.



Password selection

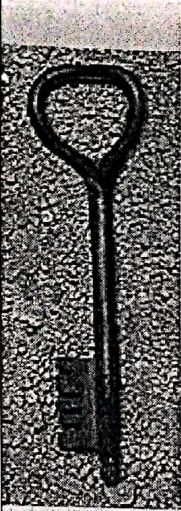
- ◆ User selection of passwords is typically very poor
 - Lower entropy password makes dictionary attacks easier
- ◆ Typical passwords:
 - Derived from account names or usernames
 - Dictionary words, reversed dictionary words, or small modifications of dictionary words
- ◆ Users typically use the same password for multiple accounts
 - Weakest account determines the security!



Better password selection

- ◆ Non-alphanumeric characters
- ◆ Longer phrases
- ◆ Can try to enforce good password selection...
- ◆ ...but these types of passwords are difficult for people to memorize and type!

Dictionary Attack – some numbers

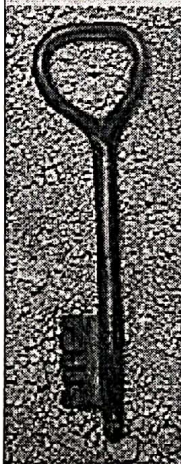


◆ Typical password dictionary

- 1,000,000 entries of common passwords
 - people's names, common pet names, and ordinary words.
- Suppose you generate and analyze 10 guesses per second
 - This may be reasonable for a web site; offline is *much* faster
- Dictionary attack in at most 100,000 seconds = 28 hours, or 14 hours on average

◆ If passwords were random

- Assume six-character password
 - Upper- and lowercase letters, digits, 32 punctuation characters
 - 94^6 689,869,781,056 password combinations.
 - Exhaustive search requires 1,093 years on average



Password-based protocols

◆ Any password-based protocol is potentially vulnerable to an “on-line” dictionary attack

- On-line attacks can be detected and limited

◆ How?

- “Three strikes”
- Ratio of successful to failed logins
- Gradually slow login response time

◆ Potential DoS

- Cache IP address of last successful login



From passwords to keys?

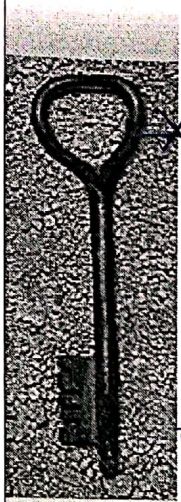
- ◆ Can potentially use passwords to derive symmetric or public keys
- ◆ What is the entropy of the resulting key?

not as secure as the AES key 128 bit



Password-based protocols

- ◆ Off-line attacks can never be 'prevented', but protocols can be made secure against such attacks
→ because he has downloaded hash file!
- ◆ Any password-based protocol is vulnerable to off-line attack if the server is compromised
 - Once the server is compromised, why do we care?



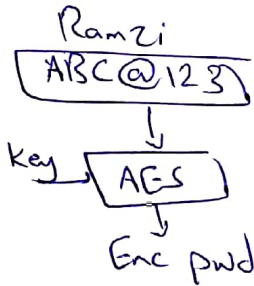
تخزين كلمة السر

Password storage

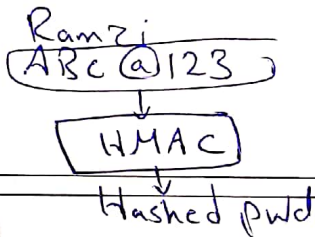
clear text
Encrypted
hashed

- ◆ "Salt"-ed hash of password
 - Makes dictionary attacks harder,
 - Prevents using 'rainbow tables'

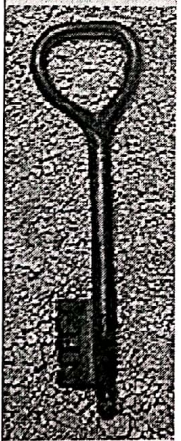
* If I stored the password Encrypted, the Encryption algorithm is reversible anyone with the Key can know the password!



* So we store it hashed, because hash is ~~one~~ one way function "not reversible"



* شرح في اللمعة



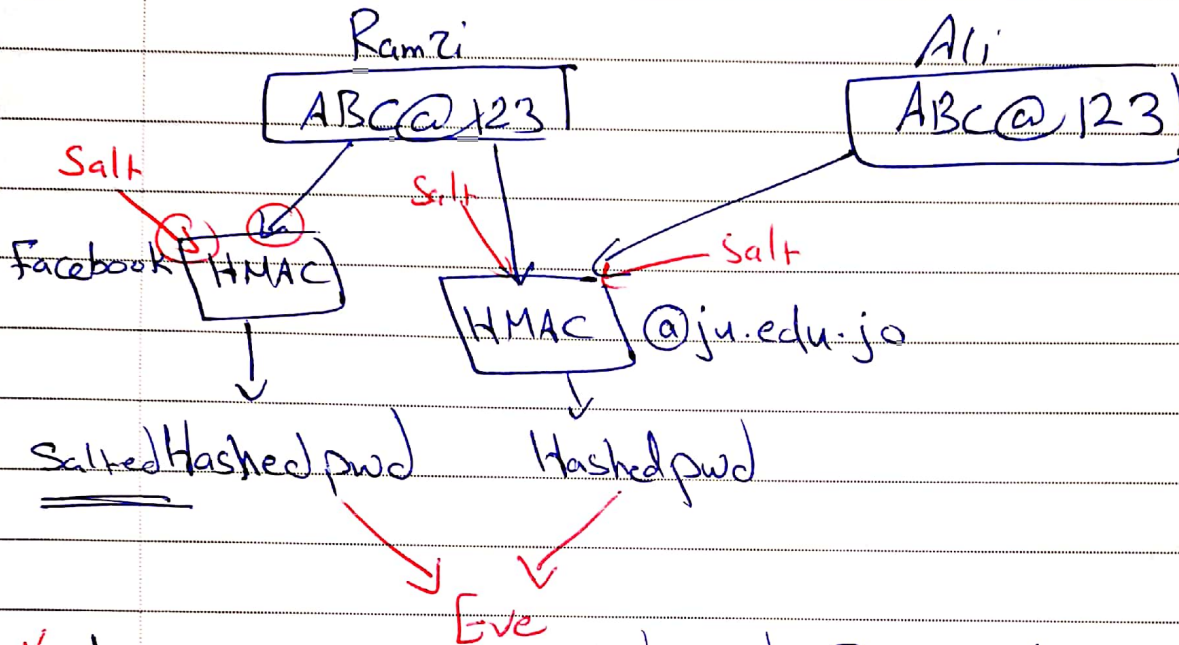
Advantages of salt

→ not Secret Value

- ◆ Without salt
 - Same hash functions on all machines
 - Compute hash of all common strings once
 - Compare hash file with all known password files
- ◆ With 12 bits salt
 - One password hashed 2^{12} different ways
 - Precompute hash file?
 - Need much larger file to cover all common strings
 - Dictionary attack on known password file
 - For each salt found in file, try all common strings

* Rainbow table: hash of all possible passwords
→ use salt to prevent Rainbow table & Dictionary attacks

* password storage & Advantages of salt



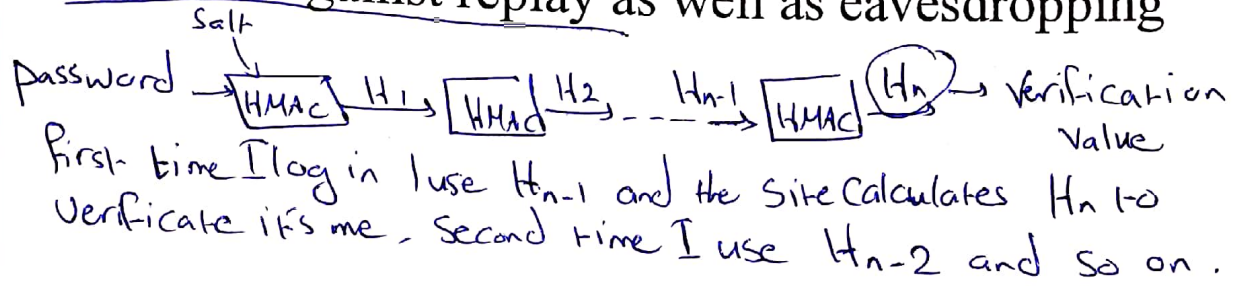
* by comparing 2 Hashed pwd Eve at least can know that Ramzi use the same password for 2 different websites. but she can't know the password.

* with salt Eve can't know that Ramzi uses the same password for 2 different website because the hashed pwd for each one will be different than other. Also even if Ali & Ramzi have the same password, the hashed value will differ because of the salt.

One-time password

- ◆ New password obtained by passing user-password through one-way function n times which keeps incrementing

- ◆ Protects against replay as well as eavesdropping



Password Cracking

Dictionary attacks

- Develop a large dictionary of possible passwords and try each against the password file
- Each password must be hashed using each salt value and then compared to stored hash values

Rainbow table attacks

- Pre-compute tables of hash values for all salts
- A mammoth table of hash values
- Can be countered by using a sufficiently large salt value and a sufficiently large hash length

Password crackers exploit the fact that people choose easily guessable passwords

- Shorter password lengths are also easier to crack

John the Ripper

- Open-source password cracker first developed in 1996
- Uses a combination of brute-force and dictionary techniques

You guess 10000 guesses / second on ~~10~~ ¹⁰ guesses ?
How many seconds you need

$$\frac{10^{10}}{10^4} = 10^6 \text{ seconds}$$

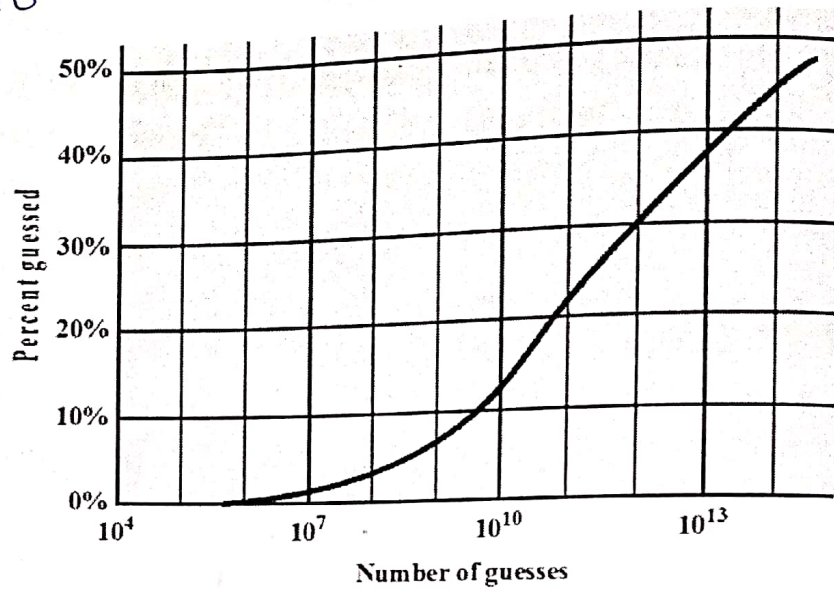
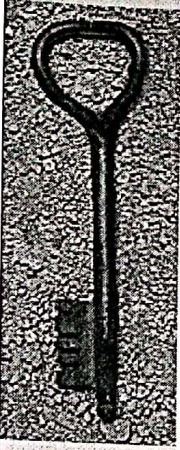
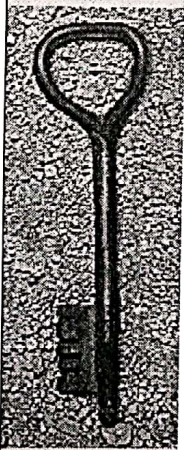


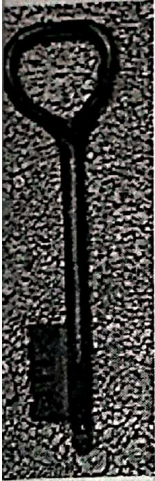
Figure 3.4 The Percentage of Passwords Guessed After a Given Number of Guesses



Passwords

Improving Security

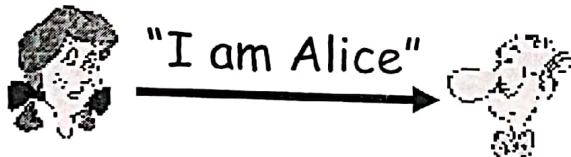
- **Password complexity**
 - Case-sensitivity
 - Use of special characters, numbers, and both upper and lower-case letters
 - Minimum length requirements
- **Security questions**
 - Ask personal questions which need to be verified
 - Some questions are very easy to discover answers
- **Virtual keyboard**
 - Person clicks on-screen keyboard to enter password (prevents keylogging)



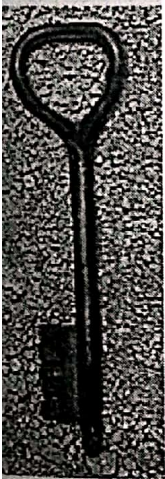
Challenge-response Authentication

Goal: Bob wants Alice to “prove” her identity to him

Protocol ap1.0: Alice says “I am Alice”



Failure scenario??



Authentication

Goal: Bob wants Alice to “prove” her identity to him

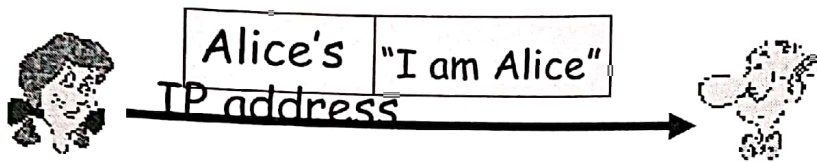
Protocol ap1.0: Alice says “I am Alice”



in a network,
Bob can not “see”
Alice, so Trudy simply
declares
herself to be Alice

Authentication: another try

Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address

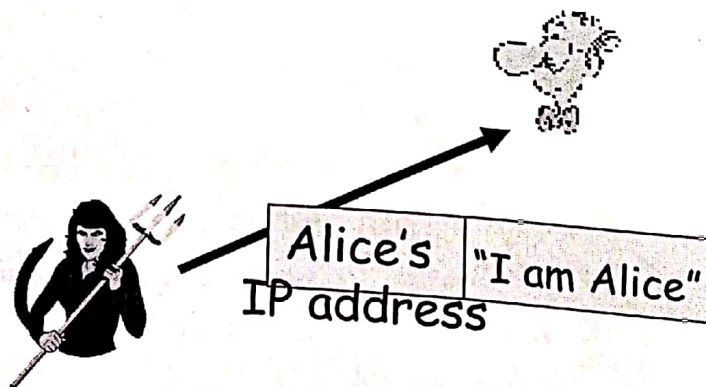


Failure scenario??



Authentication: another try

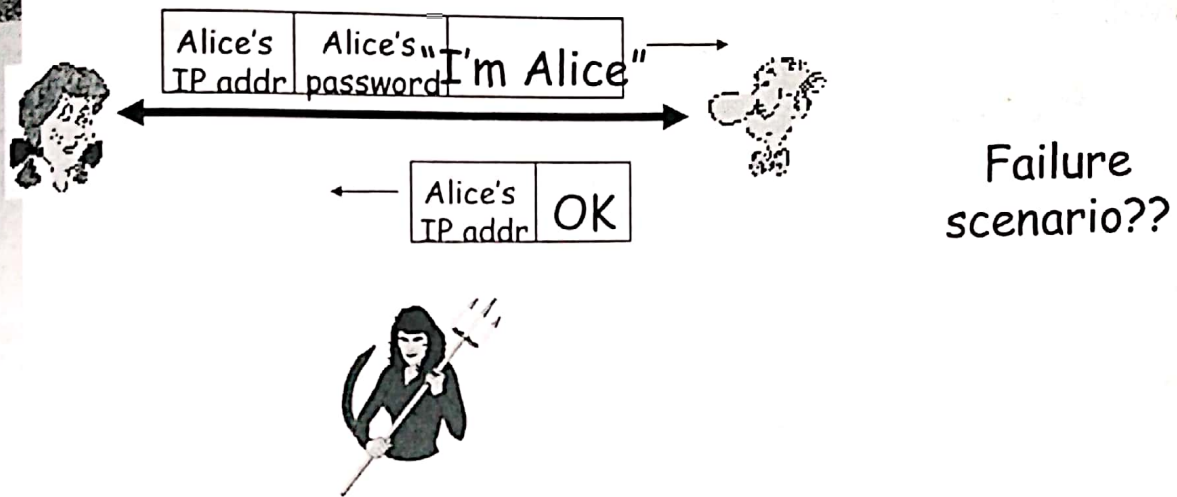
Protocol ap2.0: Alice says "I am Alice" in an IP packet containing her source IP address



Trudy can create a packet "spoofing" Alice's address

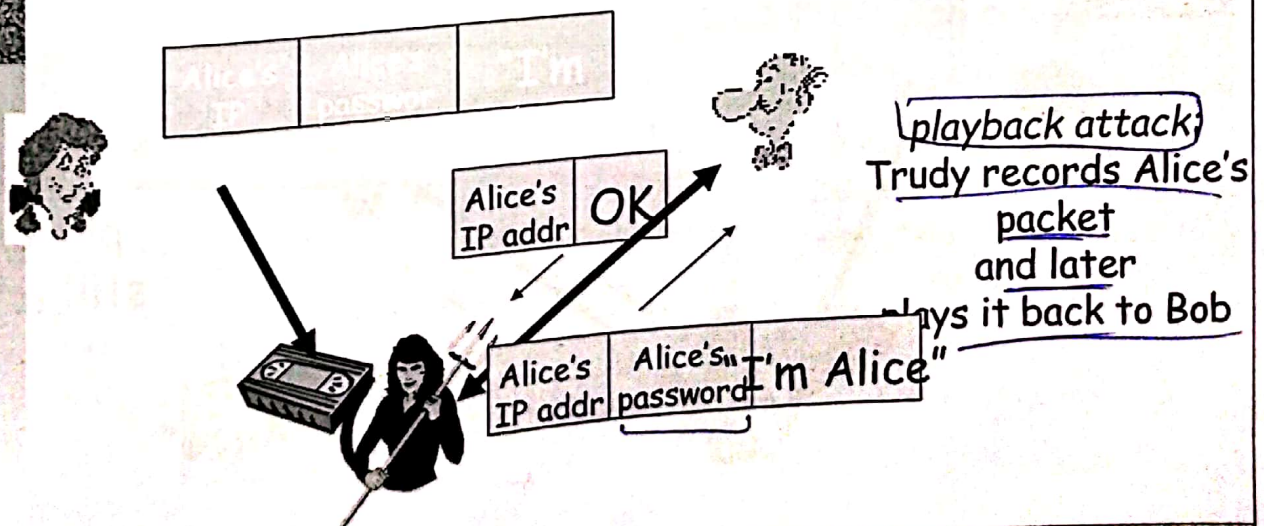
Authentication: another try

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



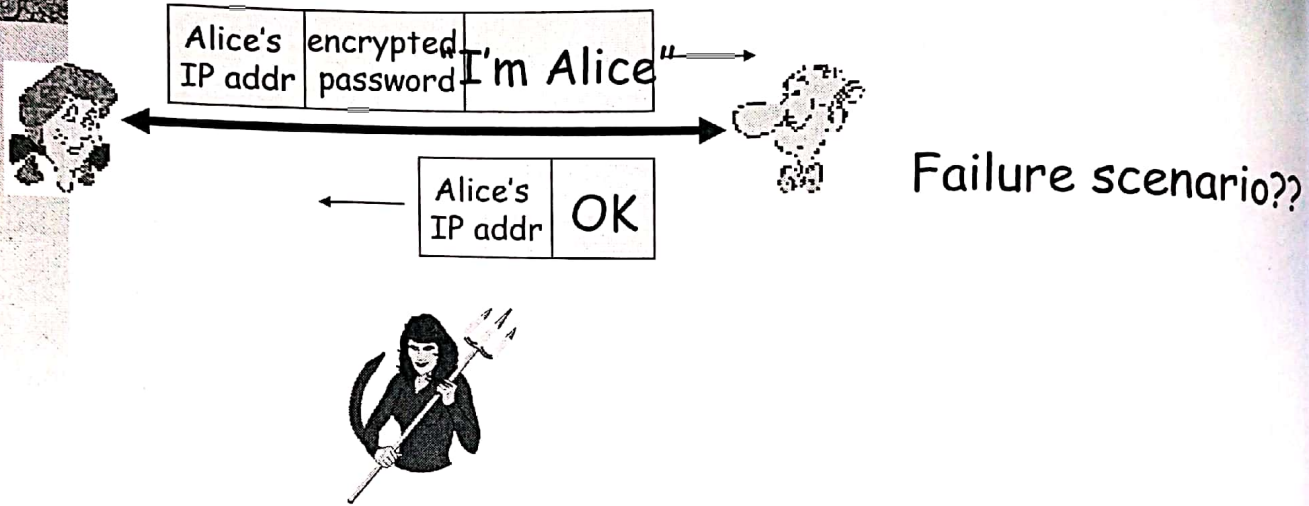
Authentication: another try

Protocol ap3.0: Alice says "I am Alice" and sends her secret password to "prove" it.



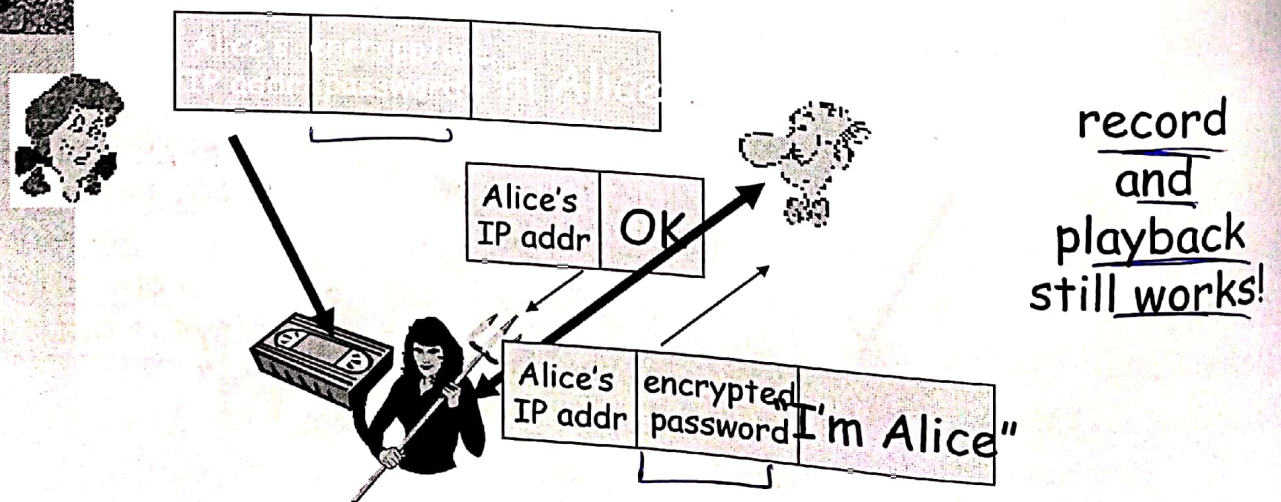
Authentication: yet another try

Protocol ap3.1: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.



Authentication: another try

Protocol ap3.1: Alice says "I am Alice" and sends her encrypted secret password to "prove" it.



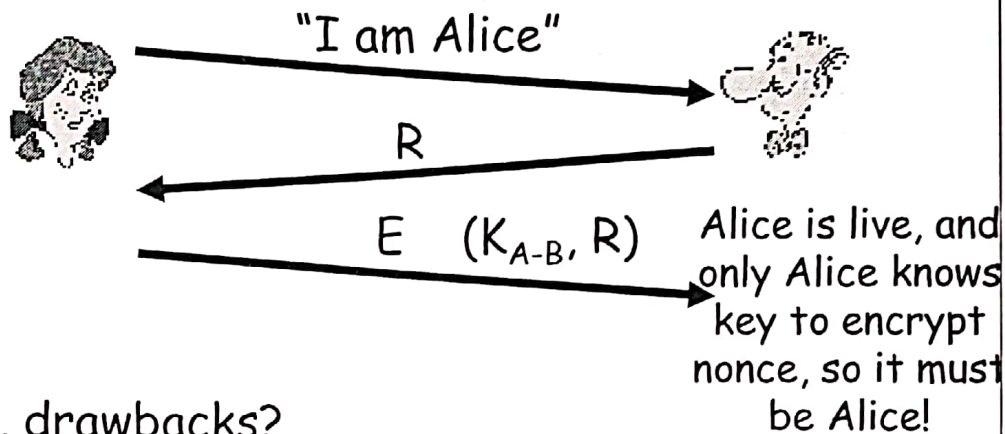
Challenge Response

Authentication: yet another try

Goal: avoid playback attack

Nonce: number (R) used only once -in-a-lifetime

ap4.0: to prove Alice "live", Bob sends Alice nonce, R. Alice must return R, encrypted with shared secret key



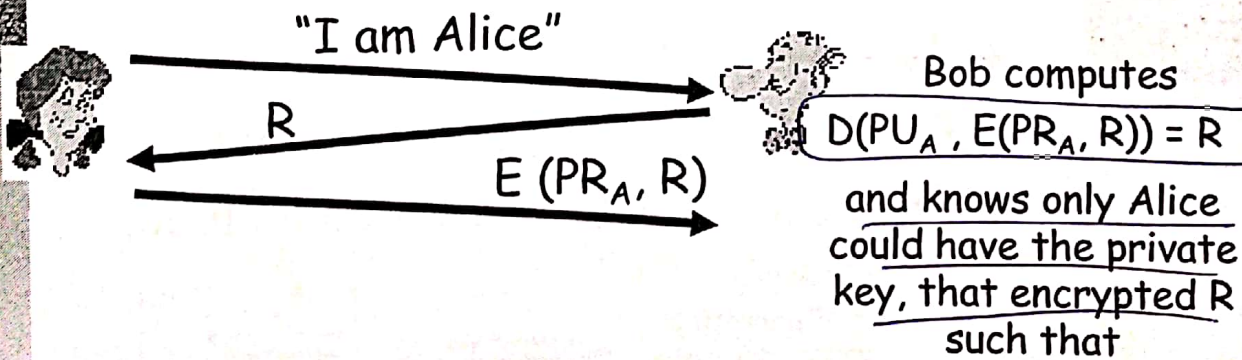
Failures, drawbacks?

Authentication: ap5.0

ap4.0 doesn't protect against server database reading

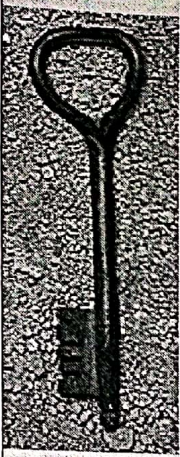
◆ can we authenticate using public key techniques?

ap5.0: use ^①nonce, ^②public key cryptography



Something you are!

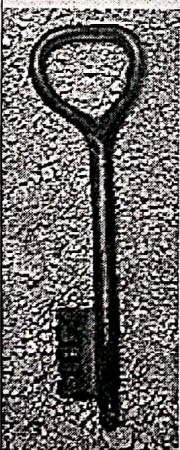
Biometrics



- ◆ Use a person's physical characteristics
 - fingerprint, voice, face, ...
- ◆ Advantages
 - Cannot be disclosed, lost, forgotten
- ◆ Disadvantages
 - Cost, installation, maintenance
 - Reliability of comparison algorithms
 - False positive; Allow access to unauthorized person
 - False negative; Disallow access to authorized person
 - Privacy? *your location will be known*
 - If forged, how do you revoke?



Biometric Authentication



- ◆ Attempts to authenticate an individual based on unique physical characteristics
- ◆ Based on pattern recognition → *features*
- ◆ Is technically complex and expensive when compared to passwords and tokens
- ◆ Physical characteristics used include:
 - Facial characteristics
 - Fingerprints
 - Hand geometry
 - Retinal pattern
 - Iris
 - Signature
 - Voice

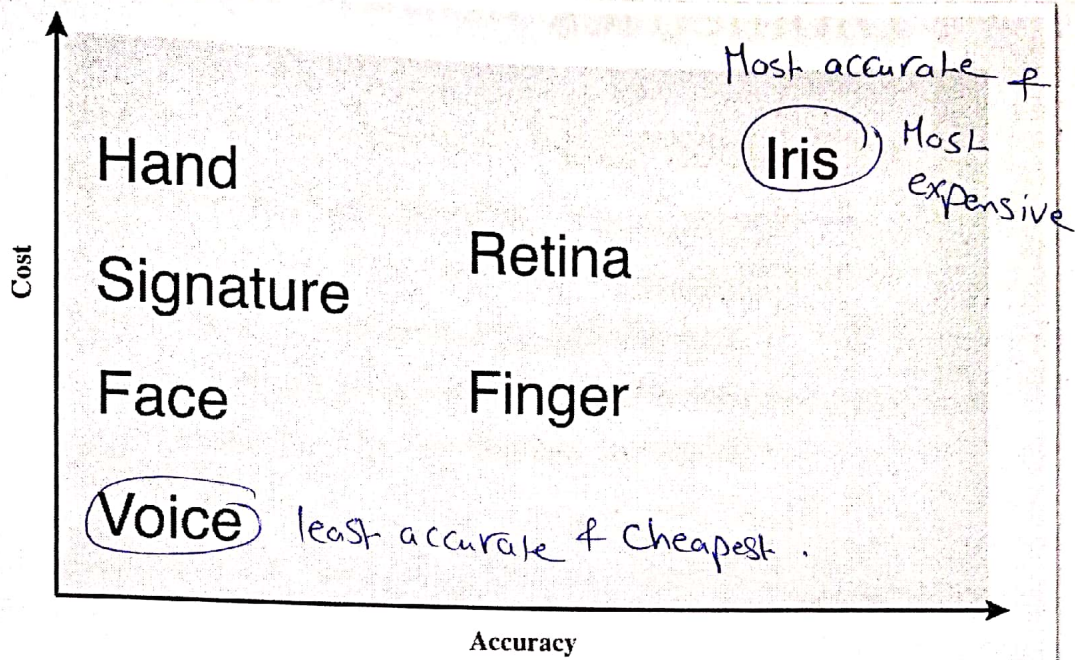
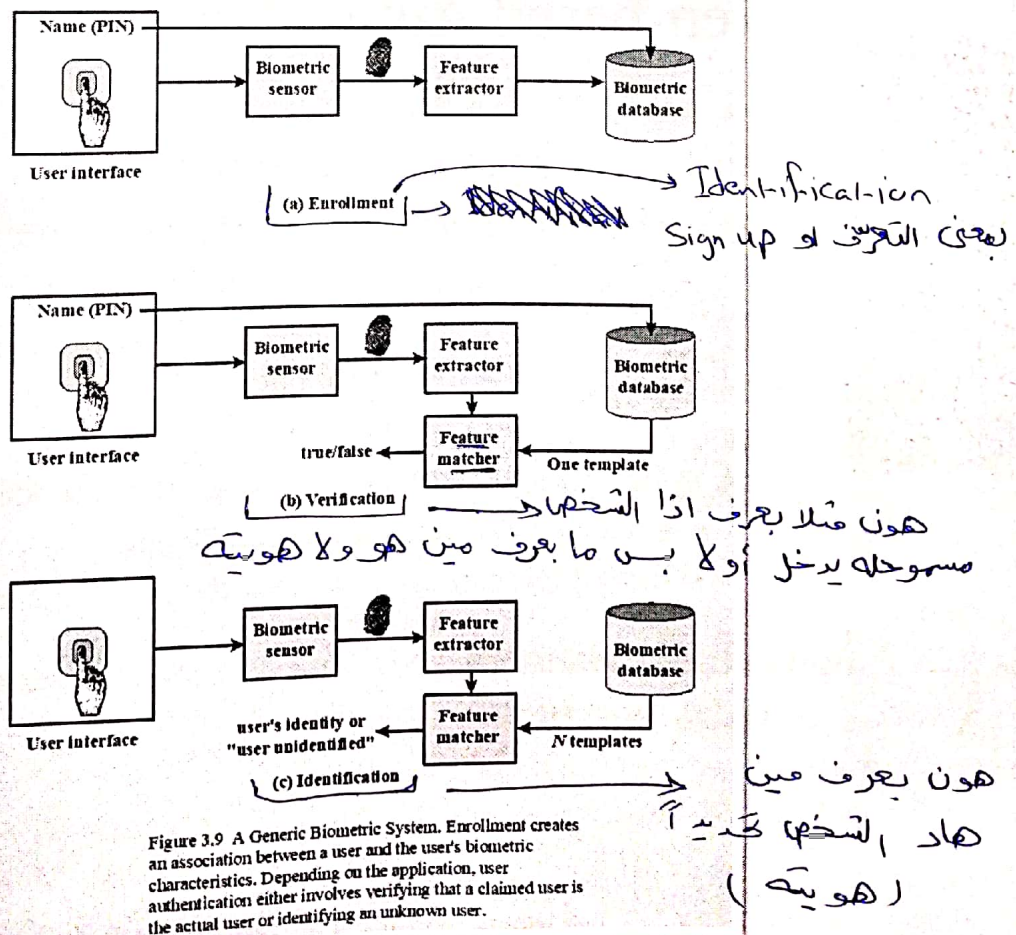


Figure 3.8 Cost Versus Accuracy of Various Biometric Characteristics in User Authentication Schemes.



Biometrics

◆ Common uses

- Specialized situations, physical security
- Combine
 - Multiple biometrics
 - Biometric and PIN
 - Biometric and token

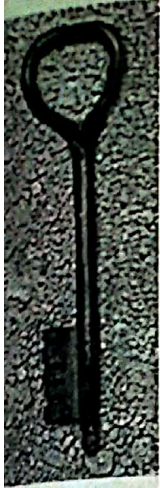


Token-based Authentication Smart Card

- ◆ With embedded CPU and memory
 - Carries conversation w/ a small card reader
- ◆ Various forms
 - PIN protected memory card
 - Enter PIN to get the password
 - Cryptographic challenge/response cards
 - Computer create a random challenge
 - Enter PIN to encrypt/decrypt the challenge w/ the card



Key Distribution



◆ given parties A and B have various **key distribution** alternatives:

انوس اسونه اعطيه باه

1. A can select key and physically deliver to B
2. third party can select & deliver key to A & B → must be trusted
3. if A & B have communicated previously can use previous key to encrypt a new key
4. if A & B have secure communications with a third party C, C can relay key between A & B
5. Diffie Hellman .

مفروضين
نفا لفتر
one
one

2-5 Using public key encryption

Trusted Intermediaries → TTP (trusted third party)



Symmetric key problem:

Public key problem:

◆ How do two entities establish shared secret key over network?

◆ When Alice obtains Bob's public key (from web site, e-mail, diskette), how does she know it is Bob's public key, not Trudy's?

Solution:

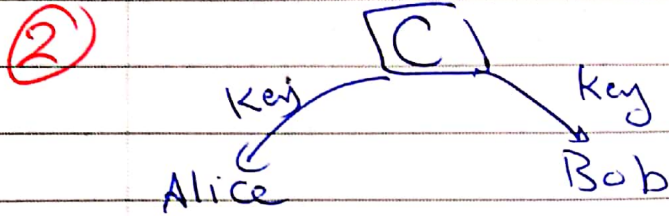
◆ trusted key distribution center (KDC) acting as intermediary between entities

Solution:

◆ trusted certification authority (CA)

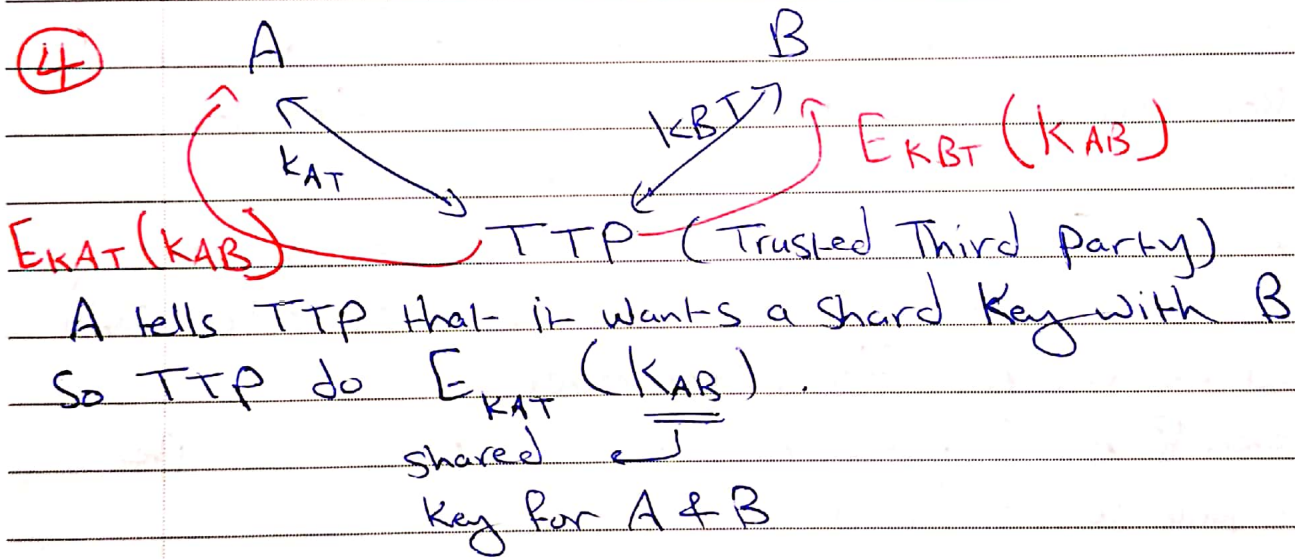
* Key Distribution :

① Alice — Bob exchange the key in Corner!



③

A	B
Key old = Kold	Key old = Kold
$E_{Kold}(K_{new})$	→



⑤ D. H (a, p, g) Man in the middle attack

$A = g^a \text{ mod } p$

$B = g^b \text{ mod } p$

$A^b \text{ mod } p$

$B^a \text{ mod } p$

No.

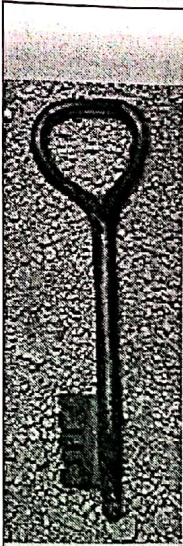
Key Distribution:

⑥ Using Public Key

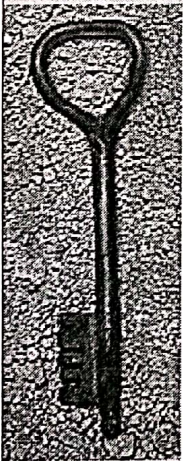
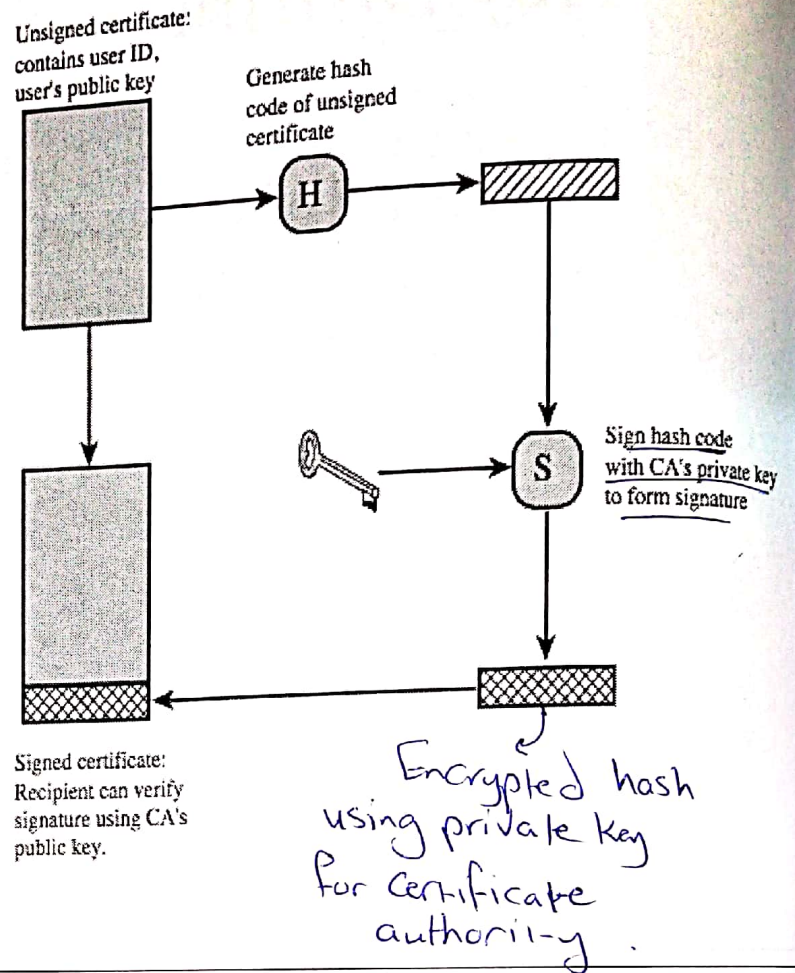
Alice = E_{PUB} (K_{AB}) → Bob

Public Key of ~~Bob~~ Bob, Bob decrypt the key sent with his private key

* needs certificate!



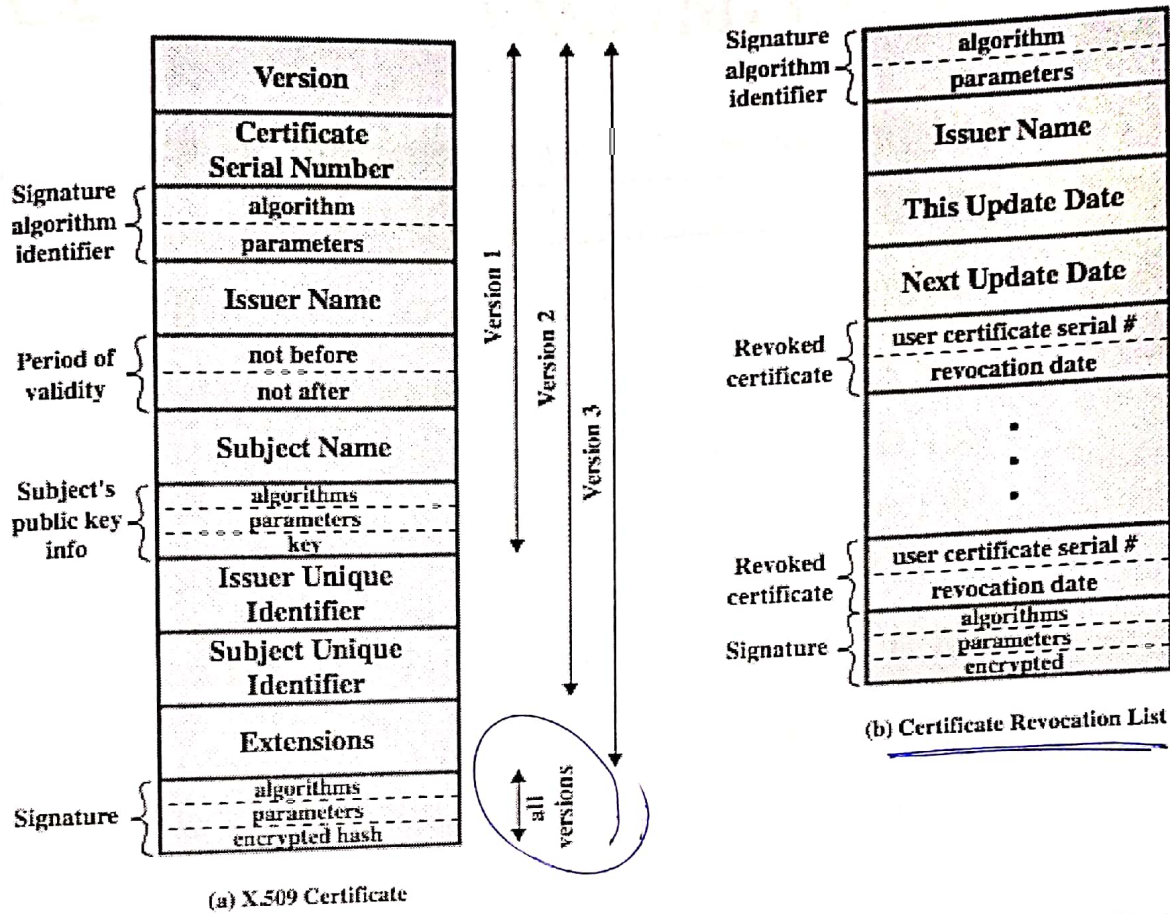
X.509 Certificate Use



X.509 Certificates

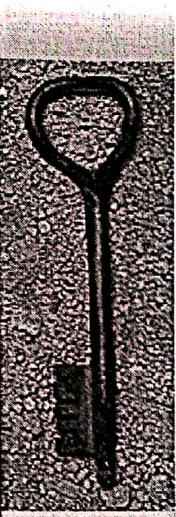
- ◆ issued by a Certification Authority (CA), containing:
 - version V (1, 2, or 3)
 - serial number SN (unique within CA) identifying certificate
 - signature algorithm identifier AI
 - issuer (X.500 name CA)
 - period of validity TA (from - to dates)
 - subject X.500 name A (name of owner)
 - subject public-key info Ap (algorithm, parameters, key)
 - issuer unique identifier (v2+)
 - subject unique identifier (v2+) *Version 2*
 - extension fields (v3)
 - signature (of hash of all fields in certificate)
- ◆ notation CA<<A>> denotes certificate for A signed by CA

X.509 Certificates



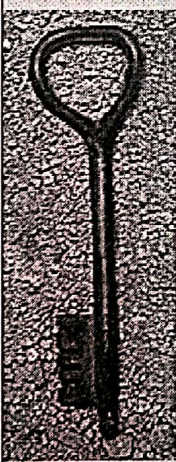
Obtaining a Certificate

- any user with access to CA can get any certificate from it
- only the CA can modify a certificate
- because cannot be forged, certificates can be placed in a public directory

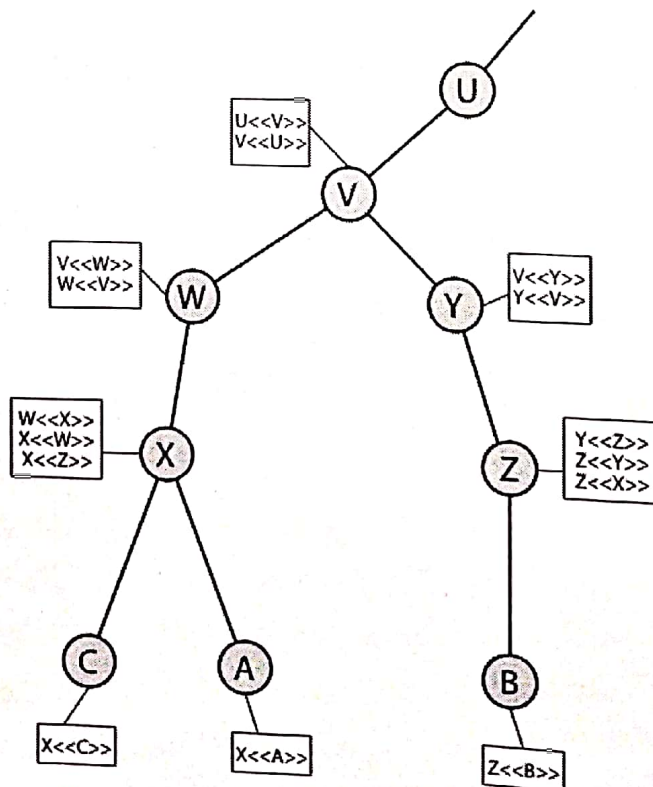


CA Hierarchy

- if both users share a common CA then they are assumed to know its public key
- otherwise CA's must form a hierarchy
- use certificates linking members of hierarchy to validate other CA's
 - each CA has certificates for clients (forward) and parent (backward)
- each client trusts parents certificates
- enable verification of any certificate from one CA by users of all other CAs in hierarchy

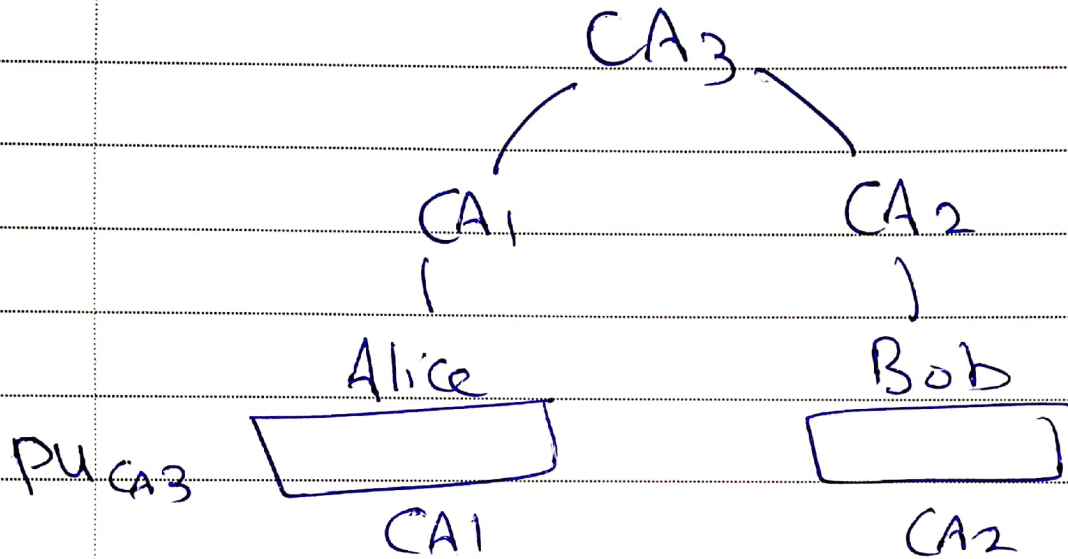


CA Hierarchy Use



* CA Hierarchy

Called =
→ Web Trust =



Alice needs to verificate Bob but she needs to know PU_{CA2} and she doesn't have it, So she will verifacate CA_2 using PU_{CA3} and then she will have PU_{CA2}

Certificate Revocation

- ◆ certificates have a period of validity
- ◆ may need to revoke before expiry, eg:
 1. user's private key is compromised
 2. user is no longer certified by this CA
 3. CA's certificate is compromised
- ◆ CA's maintain list of revoked certificates
 - the Certificate Revocation List (CRL)
- ◆ users should check certificates with CA's CRL

Kerberos

→ User Authentication ^{technique} ~~technique~~

* Very efficient & useful.

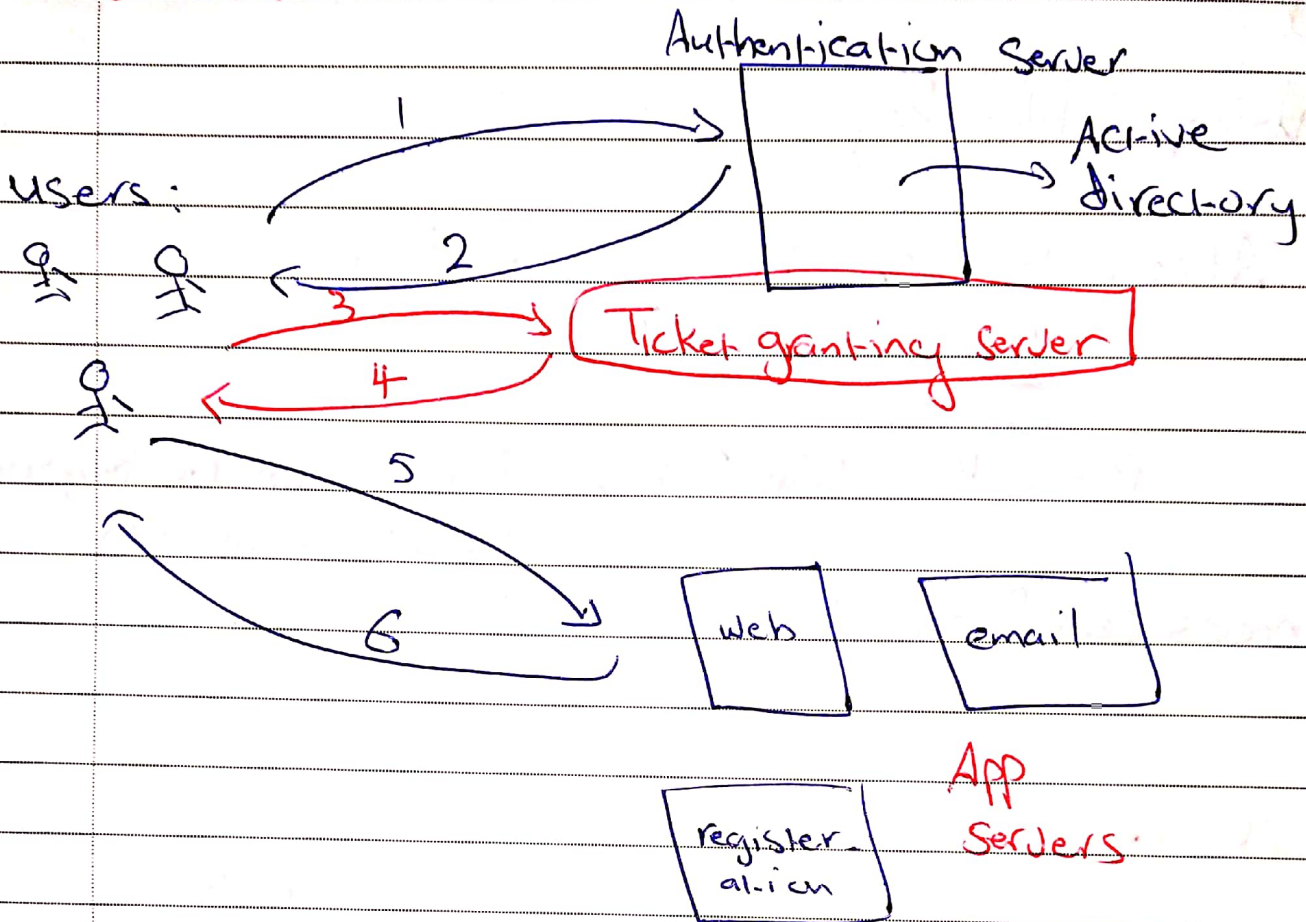
- trusted key server system from MIT
- provides centralised private-key third-party authentication in a distributed network
 - allows users access to services distributed through network
 - without needing to trust all workstations
 - rather all trust a central authentication server

➤ two versions in use: 4 & 5

* active directory like a data base for usernames & passwords.

كشرح على
النت

* Kerberos:



- * All clients have shared password with Auth server
- * All app servers have shared password with Ticket-granting server Not the Auth server

Kerberos v4 Overview

➤ a basic third-party authentication scheme

✧ have an Authentication Server (AS)

- users initially negotiate with AS to identify self
- AS provides a non-corruptible authentication credential (ticket granting ticket TGT)

✧ have a Ticket Granting server (TGS)

- users subsequently request access to other services from TGS on basis of users TGT

➤ using DES

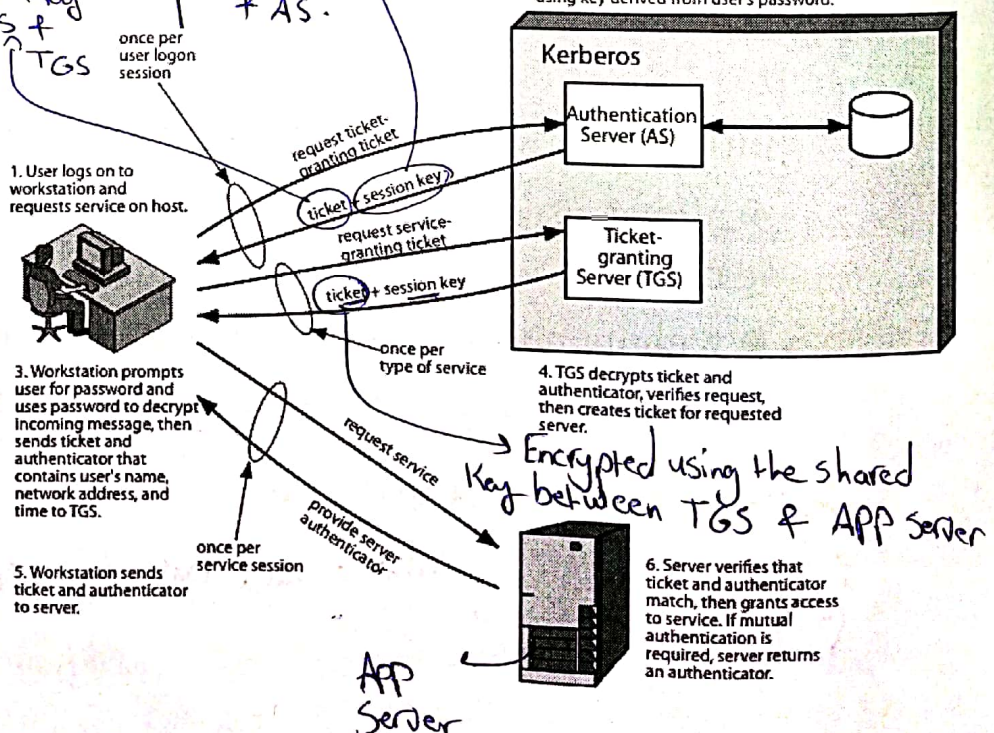
App servers بخليتي اقدر اتواصل مع

Kerberos 4 Overview

You can't read the ticket, Encrypted using the shared key between AS + TGS

Encrypted within the key shared between user + AS.

2. AS verifies user's access right in database, creates ticket-granting ticket and session key. Results are encrypted using key derived from user's password.



Kerberos v4 Dialogue

Client

(1) $C \rightarrow AS \ ID_C \parallel ID_{TGS} \parallel TS_1$
 (2) $AS \rightarrow C \ E(K_{C,AS}, [K_{C,TGS} \parallel ID_{TGS} \parallel TS_2 \parallel Lifetime_2 \parallel Ticket_{TGS}])$
 $Ticket_{TGS} = E(K_{TGS}, [K_{C,TGS} \parallel ID_C \parallel AD_C \parallel ID_{TGS} \parallel TS_2 \parallel Lifetime_2])$

(a) Authentication Service Exchange to obtain ticket-granting ticket

(3) $C \rightarrow TGS \ ID_V \parallel Ticket_{TGS} \parallel Authenticator_C$
 (4) $TGS \rightarrow C \ E(K_{C,TGS}, [K_{C,V} \parallel ID_V \parallel TS_4 \parallel Ticket_V])$
 $Ticket_{TGS} = E(K_{TGS}, [K_{C,TGS} \parallel ID_C \parallel AD_C \parallel ID_{TGS} \parallel TS_2 \parallel Lifetime_2])$
 $Ticket_V = E(K_V, [K_{C,V} \parallel ID_C \parallel AD_C \parallel ID_V \parallel TS_4 \parallel Lifetime_4])$
 $Authenticator_C = E(K_{C,TGS}, [ID_C \parallel AD_C \parallel TS_3])$

(b) Ticket-Granting Service Exchange to obtain service-granting ticket

(5) $C \rightarrow V \ Ticket_V \parallel Authenticator_C$
 (6) $V \rightarrow C \ E(K_{C,V}, [TS_5 + 1])$ (for mutual authentication)
 $Ticket_V = E(K_V, [K_{C,V} \parallel ID_C \parallel AD_C \parallel ID_V \parallel TS_4 \parallel Lifetime_4])$
 $Authenticator_C = E(K_{C,V}, [ID_C \parallel AD_C \parallel TS_5])$

(c) Client/Server Authentication Exchange to obtain service

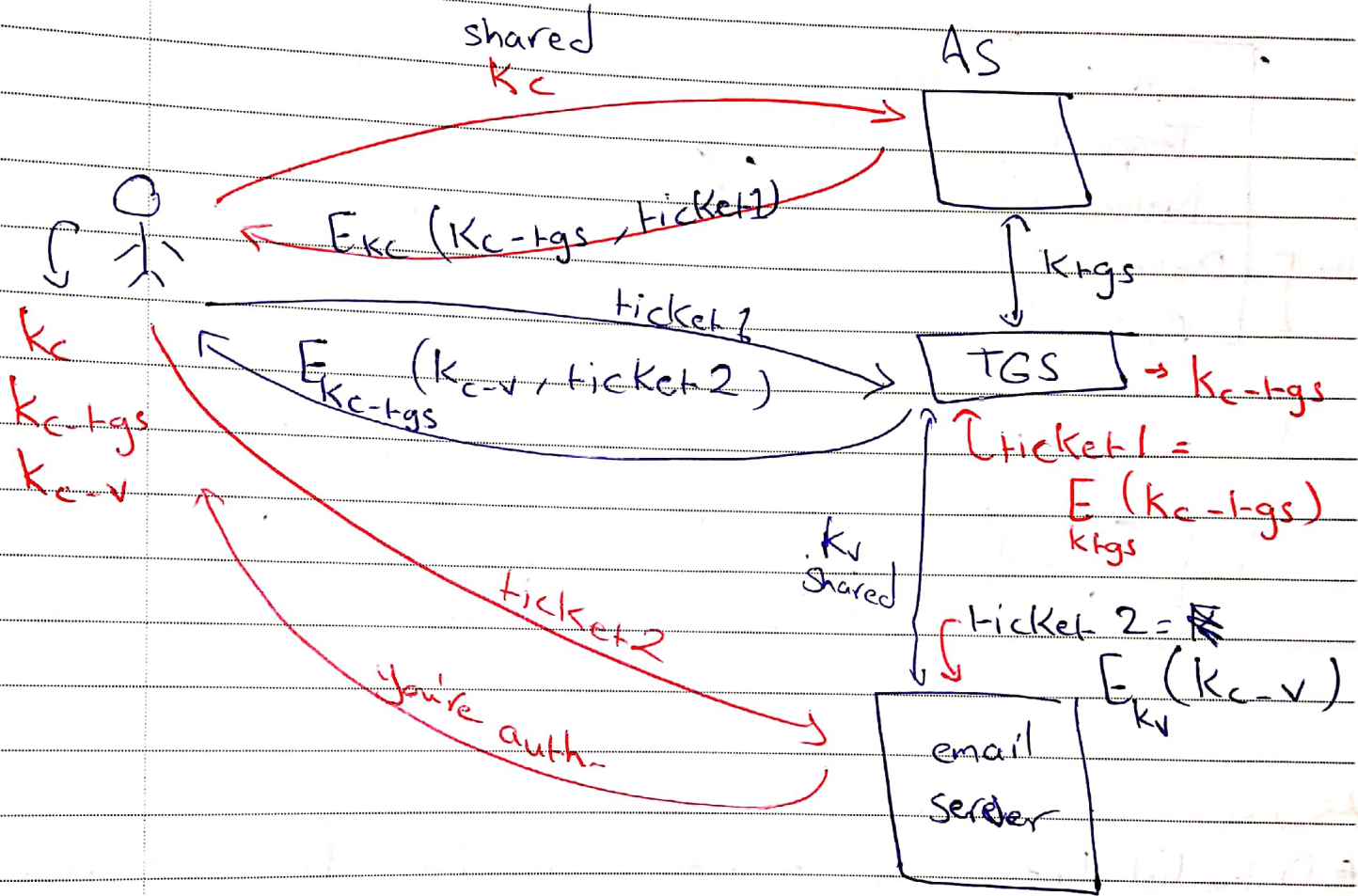
سج في
الدفتر

Kerberos Realms

- ◆ a Kerberos environment consists of:
 - a Kerberos server $\begin{matrix} \swarrow AS \\ \searrow TGS \end{matrix}$
 - a number of clients, all registered with server
 - application servers, sharing keys with server
- ◆ this is termed a realm
 - typically a single administrative domain
- ◆ if have multiple realms, their Kerberos servers must share keys and trust

like: If you login to your instagram account using your facebook account.

Kerberos V4 Dialogue :-



* Without Authentication, there will be NO Security!

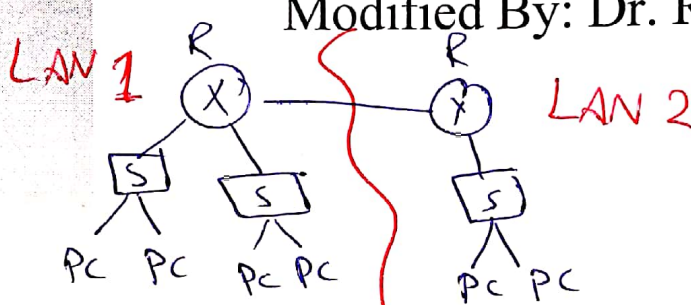
ببإذن من الإدارة
وتحت إشرافها
LAN



LAN Security

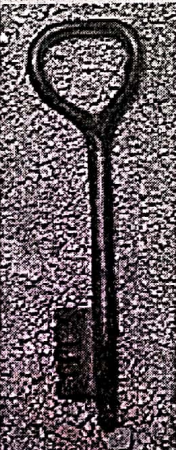
↳ Local Area Networks
(limited space)

Modified By: Dr. Ramzi Saifan



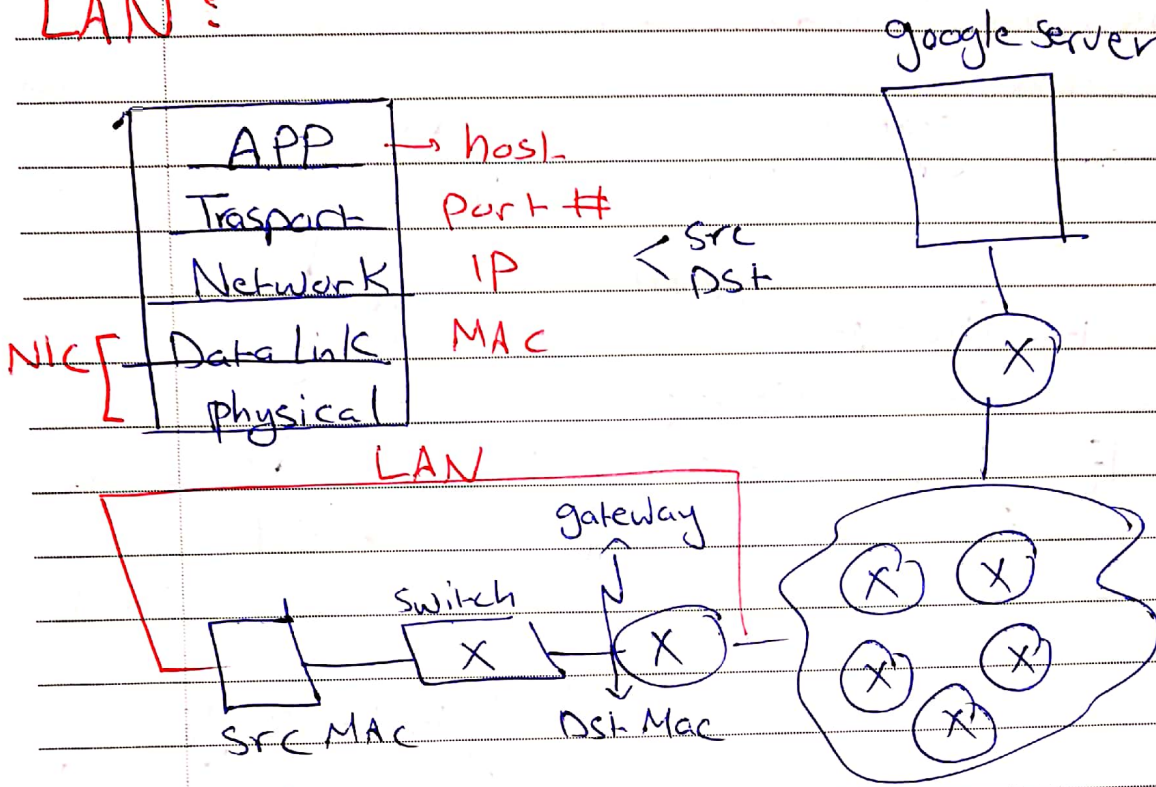
LAN

شرح في البث



- ◆ Many data traffic is available to every node in the LAN zone → Network Interface Card.
- ◆ NIC provides physical and logical conversions.
- ◆ Every NIC has an address.
- ◆ When messages are inserted in the network, the address of the destination NIC is part of the message header.
- ◆ As messages flow through an NIC, the destination address is examined.
- ◆ If the destination address matches the NIC doing the examining, the message is transmitted to upper layers.
- ◆ It is also easy to provide broadcast communication to all NICs by using a special address such as the binary value of all ones.

LAN :



*Switch is called transport device

• Data link layer gives SRC MAC & DST MAC inside a LAN only.

LAN simplicity-security tradeoff

- ◆ There are many reasons why LANs have become popular, ?

- the most important is flexibility and cost.
- New NICs may be added to the net or activated,
- or NICs may be removed or deactivated without making a significant change.
- This dynamic flexibility happens without notification and coordination with a central authority.

No need for Settings & Config.

- ◆ A PC can record all the communications traffic.

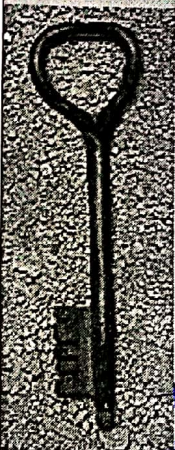
Address filtering can be turned off.

- The NIC can operate in "promiscuous" or "snooper" mode, passing all traffic to the PC, which in turn can record it for some future use.

Wiretapping

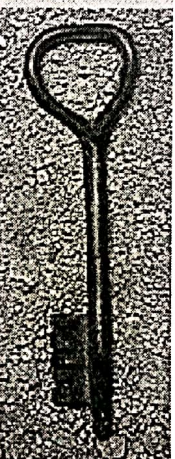
- ◆ Wiretapping is conventionally subdivided into passive and active categories.
- ◆ In passive, the message traffic is observed but not modified.
 - The most obvious objective of passive wiretapping is to learn the contents of messages;
 - traffic analysis may provide the adversary with information when message content is not available.
 - E.g., sudden change in traffic volume between national central banks, might signal a change in the rate of exchange or some other financial activity that could be turned into a profit by someone.
- ◆ In active wiretapping: Messages can be completely deleted, they can be inserted, or their contents can be modified.
 - Delay, reordering, duplication, and retransmission are also possible.

Passive attacks Wiretapping



Packet Sniffing

- ◆ This works for wireless too!
- ◆ In fact, it works for any broadcast-based medium
- * LAN is a single broadcast domain.
- * If I try to connect to a LAN I will need an IP, Subnet Mask, gateway. So I will ask the DHCP for that, then I will be able to receive anything broadcasted inside the LAN.
- * anyone in the wiretapping mode can get any message broadcasted.



Packet Sniffing Countermeasures

- ◆ How can we protect ourselves?
- ◆ SSH, not Telnet
 - Many people are still using Telnet and send their password in the clear (use PuTTY instead!) → Encrypted
 - Now that I have told you this, please do not exploit this information
 - Packet sniffing is, by the way, prohibited by Computing Services
- ◆ HTTP over SSL → Secure socket layer
 - Especially when making purchases with credit cards!
- ◆ SFTP, not FTP
 - Unless you **really** don't care about the password or data
 - Can also use KerbFTP (download from MyAndrew)
- ◆ IPsec
 - Provides network-layer confidentiality

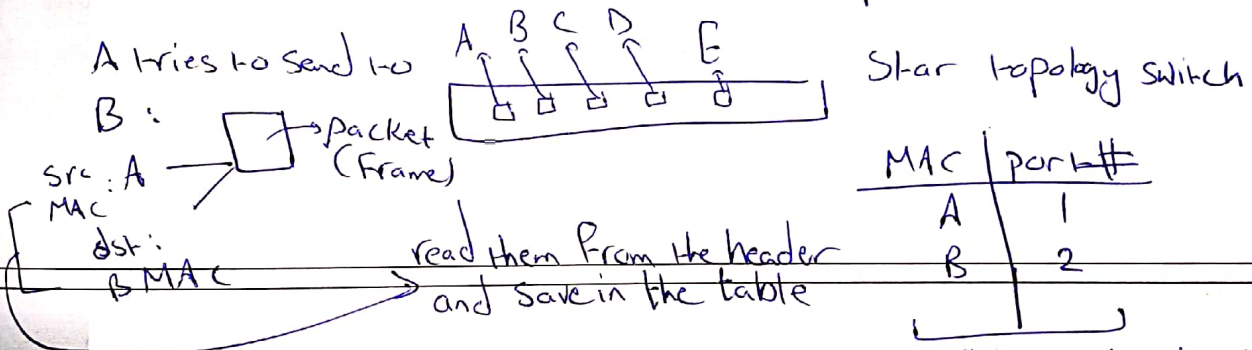
Secure

Self learning device
↑

Switch Learning Attacks

- ◆ Switch learning is what makes Ethernet scale
- ◆ Two key attacks: MAC flooding and spoofing
 - Extremely simple to carry out, yet very potent
 - Can help attacker collect usernames/passwords, prevent proper operation of LAN, etc
 - Can turn a \$50,000 switch into a \$12 hub

* Switch Learn which MAC to which port.



this table in the Switch memory

Limitations on switch memory

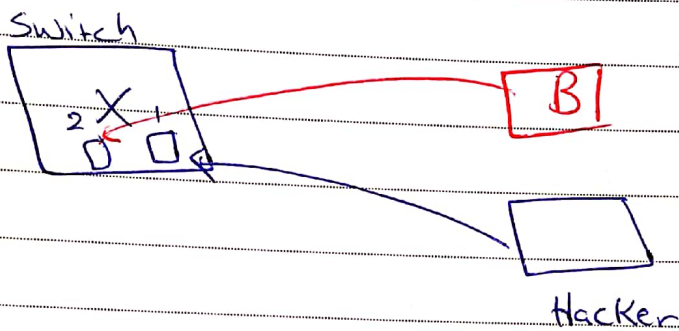
- ◆ High end switches can store hundreds of thousands of learning table entries
- ◆ What happens if learning table fills up?
- ◆ Depends on vendor -
 - ✗ - Most Cisco switches do not replace older entries with new ones.
 - Need to "age out" entries (wait for them to time out)
 - ✗ - Other switches circular buffer
 - Existing entries get overwritten

TTL
↓
time to live

لا مسح في الذاكرة

Limitations on Switch memory

1) Most Cisco switches don't replace older entries with new ones.



MAC	Port #	TTL	Storage date
A	1		
B	1		
C	1		
D	1		
E	1		

* When B wants to send to port 2, the learning table is full and this causes **MAC Flooding**

2) Other Switches Circular buffer

MAC	Port #	TTL	Storage date
M	2		
N	3		
O	4		
P	5		
Q	6		

* Suppose hacker A starts to send frames.
The table will be:

MAC	port #	TTL	Storage date
M A	2 1		
N B	3 2		
O C	4 3		
P D	5 4		
Q E	6 5		

* Replace existing MAC with New One's.

①

MAC Flooding Attack

- ◆ Problem: attacker can cause learning table to fill –
Generate many packets to varied (perhaps
nonexistent) MAC addresses
- ◆ This harms efficiency
 - Effectively transforms switch into hub
 - Wastes bandwidth, and end-host CPU
- ◆ This harms privacy
 - Attacker can eavesdrop by preventing switch from learning destination of a flow
 - Causes flow's packet to be flooded throughout LAN

②

MAC Spoofing Attack

- ◆ Host pretends to own the MAC address of another host
 - Easy to do: most Ethernet adapters allow their address to be modified
 - Powerful: can immediately cause complete DoS to spoofed host
 - All learning table entries point to the attacker
 - All traffic redirected to attacker
 - Can enable attacker to evade ACLs set based on MAC information

Denial of Service
Access Control List

* How to prevent MAC Flooding & MAC Spoofing?

Switch Learning Attacks: Countermeasures

1 Detecting MAC activity

- Many switches can be config'd to warn administrator about many sudden MAC address moves

2 Port Security

- Ties a given MAC address to a port
- On violation, can drop frames, disable port for specified duration, signal alarm, increment violation counter

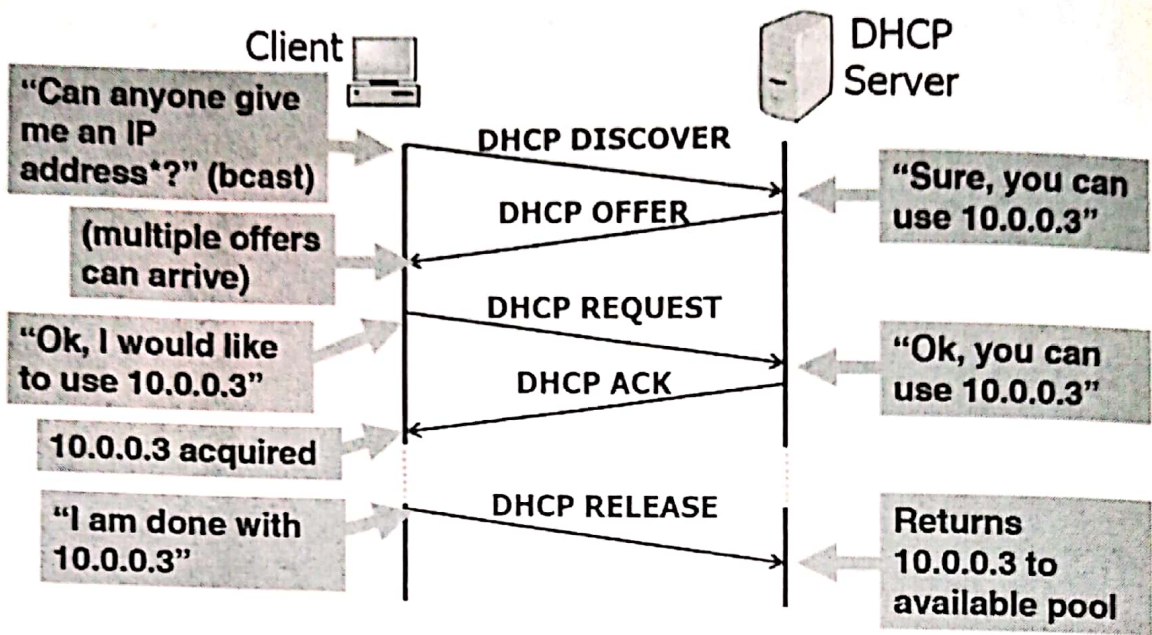
Static
Dynamic
Sticky

Switch Learning Attacks: Countermeasures

3 Unicast Flooding Protection

- Send alert when user-defined rate limit is exceeded
- Can also filter traffic or shut down port generating excessive floods

DHCP → Dynamic Host Configuration Protocol.

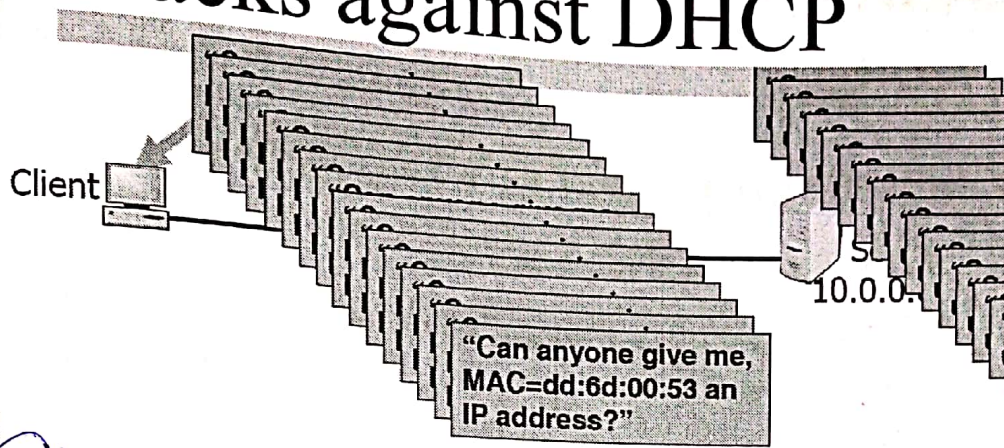


⁴³
*and other config information

Attacks on DHCP → 2 types.

- ◆ Unfortunately, DHCP was designed without security in mind
 - Whoever requests an address is free to receive one
 - No authentication fields or any other security-inclined information in protocol

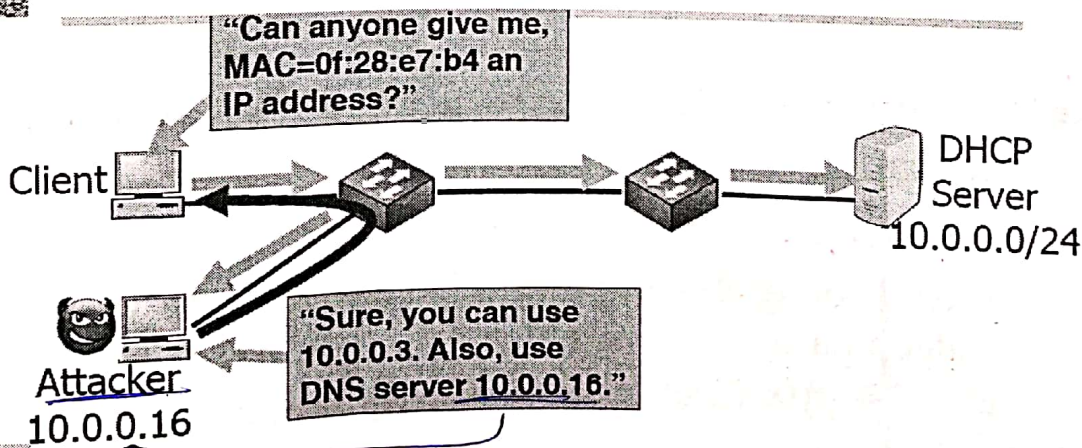
Attacks against DHCP



1. **DHCP Scope Exhaustion** → Sending many requests
- Malicious client attempts to seize entire range of IP addresses
 - When legitimate client tries, it is abandoned with no IP connectivity
- So I will have all the IP's causing in Denial of service

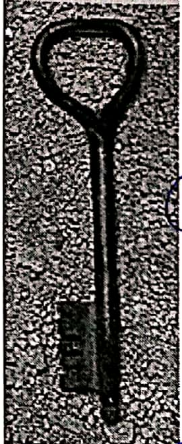
انوار 'Attacker' یعنی انو هو ال DHCP

Attack: Rogue DHCP Server



2. **Installation of a Rogue DHCP Server**
- Client uses offer or of previously-used IP address, if none then uses first-received response:
 - Rogue can compromise all clients "near" itself

Countermeasures to DHCP Attacks

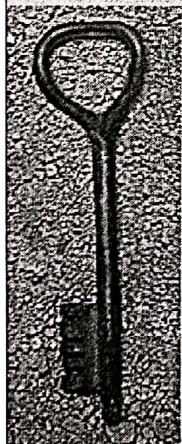


1. Limit number or set of MAC addresses per port
 - This is called Port Security
 - Limit can be set manually or switch can be instructed to lock down on first dynamically learned address

2. Limitations

- DHCP lets you request multiple IP addresses for a single MAC address

Countermeasures to DHCP Attacks

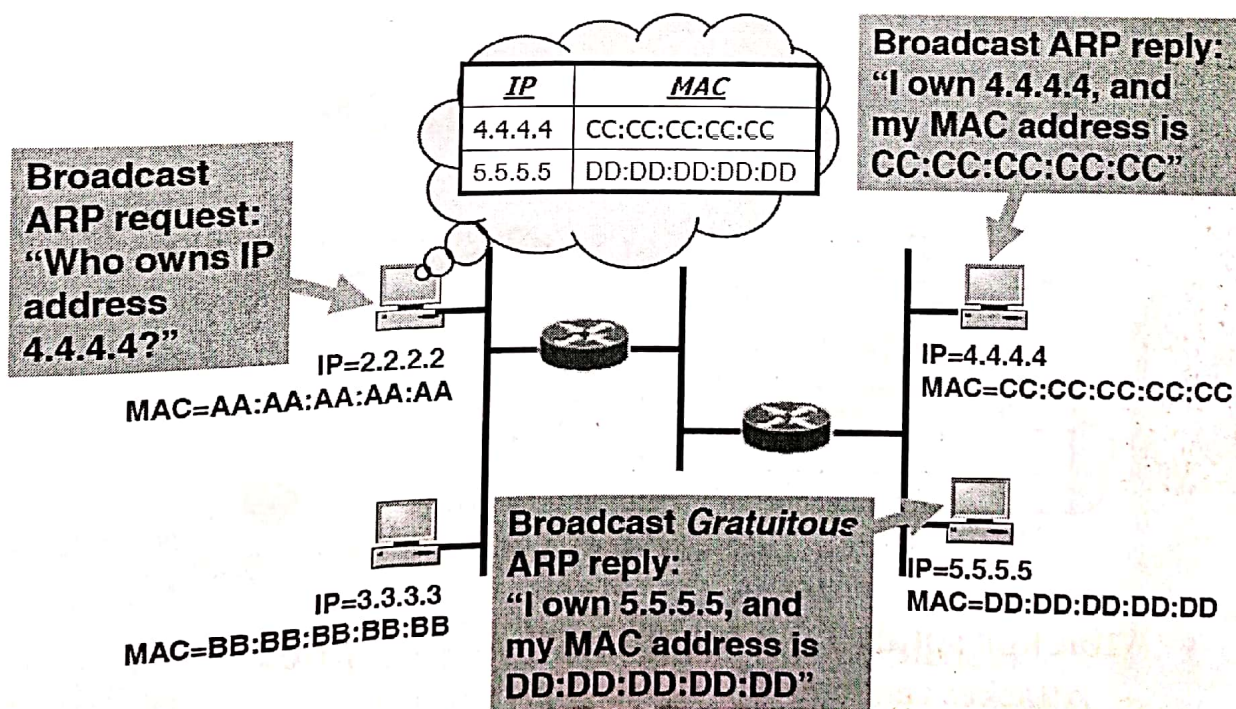


3. Prevent hosts from generating certain DHCP messages (DHCP Snooping)

- Like a stateful firewall for DHCP
- Runs on router's central management processor, to do deep packet inspection
- Learns IP-to-MAC bindings by snooping on DHCP packets
- Rules:
 - If port is connected to host, don't allow DHCP OFFER and DHCP ACK packets
 - Don't allow DHCP packets that don't match learned bindings
 - Can also rate-limit DHCP messages per port, etc

Address Resolution Protocol (ARP)

- ◆ Networked applications are programmed to deal with IP addresses
- ◆ But Ethernet forwards to MAC address
- ◆ How can OS know the MAC address corresponding to a given IP address?
- ◆ Solution: Address Resolution Protocol
 - Broadcasts ARP request for MAC address owning a given IP address



- ARP: determine mapping from IP to MAC address
- What if IP address not on subnet?
 - Each host configured with "default gateway", use ARP to resolve its IP address
- Gratuitous ARP: tell network your IP to MAC mapping
 - Used to detect IP conflicts, IP address changes; update other machines' ARP tables, update bridges' learned information

Risk Analysis for ARP

◆ No authentication

- Hosts do not sign ARP replies

◆ Information leak

- All hosts in same VLAN learn the advertised <IP,MAC> mapping
- All hosts discover querying host wishes to communicate with replying host

◆ Availability

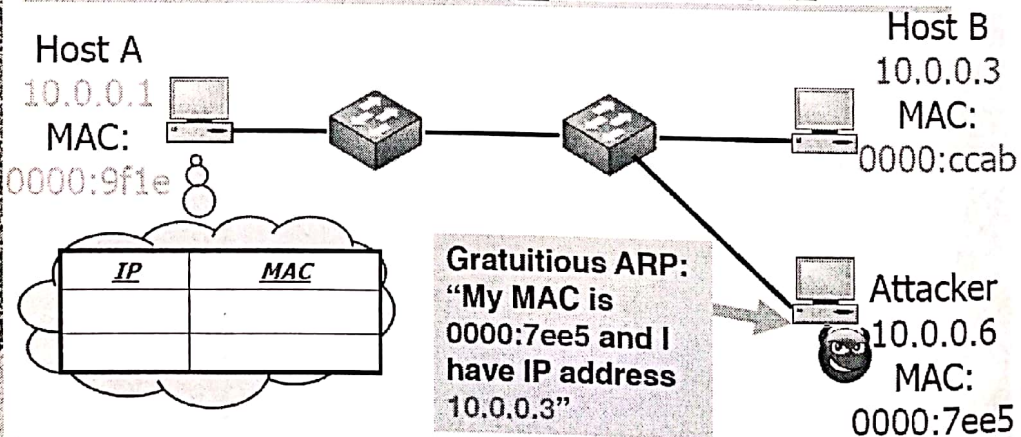
← All hosts on same LAN receive ARP request, must process it in software

- Attacker could send high rate of spurious ARP requests, overloading other hosts

all read it
if it's their
MAC they will
reply, otherwise
discard!

* DHCP to know the IP of ~~route~~ gateway.
* to learn anything inside the LAN itself use broadcast.

ARP Spoofing Attack



◆ Attacker sends fake unsolicited ARP replies

- Attacker can intercept forward-path traffic
- Can intercept reverse-path traffic by repeating attack for source
- Gratuitious ARPs make this easy
- Only works within same subnet/VLAN

فوق كل با حاد! يطلب IP معين او attacker يدعي انه ال IP انه .

← Solutions for ARP spoofing?
← How to prevent ARP spoofing attack?

Countermeasures to ARP Spoofing

- ◆ Ignore Gratuitous ARP
 - Problems: gratuitous ARP is useful, doesn't completely solve the problem
- ◆ Dynamic ARP Inspection (DAI)
 - Switches record <IP,MAC> mappings learned from DHCP messages, drop all mismatching ARP replies
- ◆ Intrusion detection systems (IDS)
 - Monitor all <IP,MAC> mappings, signal alarms

SSL and IPsec

Secure Socket
Layer

Modified by: Dr. Ramzi Saifan

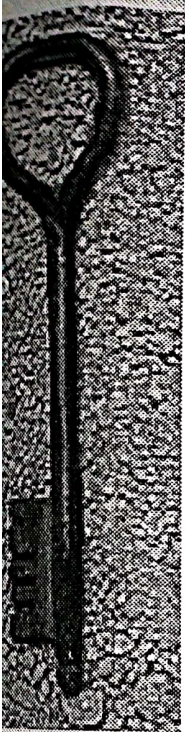
Interface between the user & Network.

Network layers

- ◆ Application protocols $\left\{ \begin{array}{l} \text{HTTP} \\ \text{SMTP} \\ \text{IP} \\ \text{DHCP} \end{array} \right.$
- ◆ Transport protocols $\left\{ \begin{array}{l} \text{UDP} \\ \text{TCP} \end{array} \right.$
- ◆ Network protocols $\left\{ \begin{array}{l} \text{IP} \\ \text{OSPF} \\ \text{RIP} \end{array} \right.$
- ◆ Data link protocols $\left\{ \begin{array}{l} \text{Ethernet} \\ \text{wifi} \end{array} \right.$
- ◆ Physical

Example security protocols

- ◆ Application layer: PGP \rightarrow pretty good privacy
- ◆ Transport layer: SSL/TLS \rightarrow Transport layer security
- ◆ Network layer: IPsec
- ◆ Data link layer: IEEE 802.11
- ◆ Security at the physical layer?



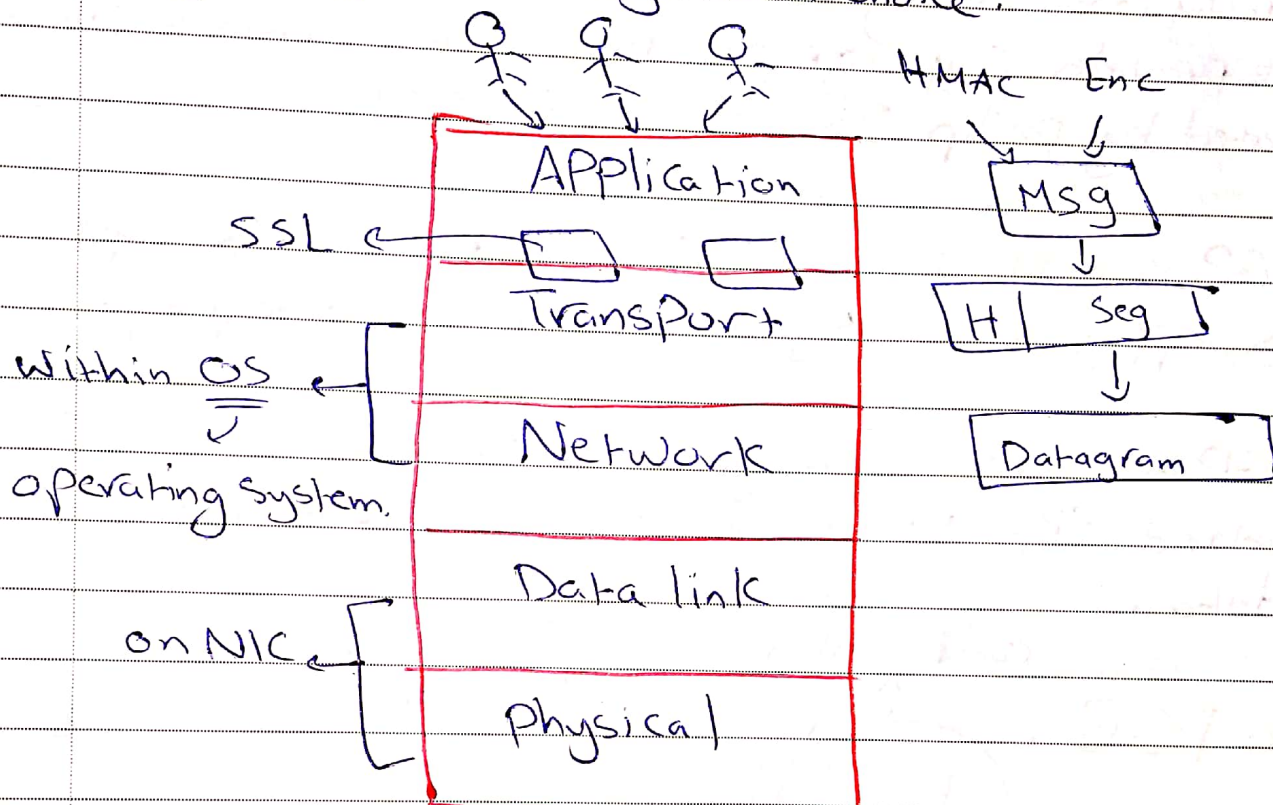
Security in what layer?

شرح في دفتر

- ◆ Depends on the purpose...
 - What information needs to be protected?
 - Who shares keys in advance?
 - Should the user be involved?
- ◆ E.g., a network-layer protocol cannot authenticate two end-users to each other
- ◆ An application-layer protocol cannot protect IP header information
- ◆ Also affects efficiency, ease of deployment, etc.

* Security in what layer?

* Security in Application layer is a choice.



* Security in Network is not a choice.

→ If the Security is in Network, you don't need to do the Security in Application layer.

* Security in Application layer only is not enough.

* If you want to do security for a user or specific application you have to do it in the Application layer because Network can't recognize users & Applications.

Generally...

- ◆ When security is placed at lower levels, it can provide automatic, "blanket" coverage...
 - ...but it can take a long time before it is widely adopted

یعنی بتعمیل Security لکھل اسے خودکار

- ◆ When security is placed at higher levels, individual users can choose when to use it...
 - ...but users who are not security-conscious may not take advantage of it



Note...

- ◆ The “best” solution is not necessarily to use PGP over IPsec!
 - Would have been better to design the Internet with security in mind from the beginning...



Example: PGP vs. SSL vs. IPsec

- ◆ PGP is an application-level protocol for “secure email”
 - Can provide security on “insecure” systems
 - Users choose when to use PGP; user must be involved
 - Alice’s signature on an email proves that Alice actually generated the message, and it was received unaltered; also non-repudiation
- ◆ In contrast, SSL would secure “the connection” from Alice’s computer;
 - would need an additional mechanism to authenticate the user
- ◆ IPsec is between every two hops in the network

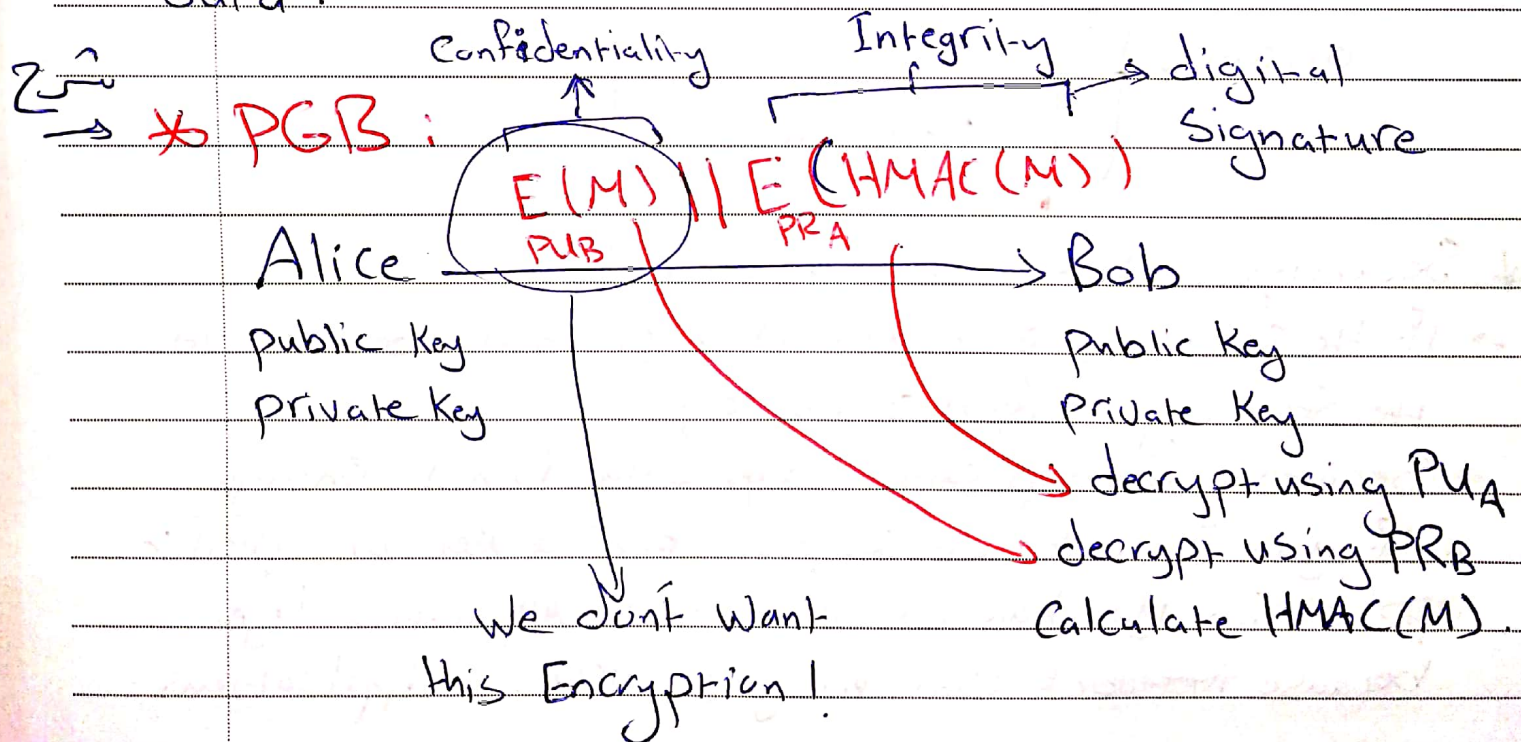
slide

→ Pretty Good Privacy (PGP)

* Pretty Good Privacy (PGP) is a Computer Program that provides cryptographic privacy and authentication. Created by Philip Zimmermann in 1991.

* PGP is a windows tool commonly used to Encrypt files, apply digital signature and enforce integrity.

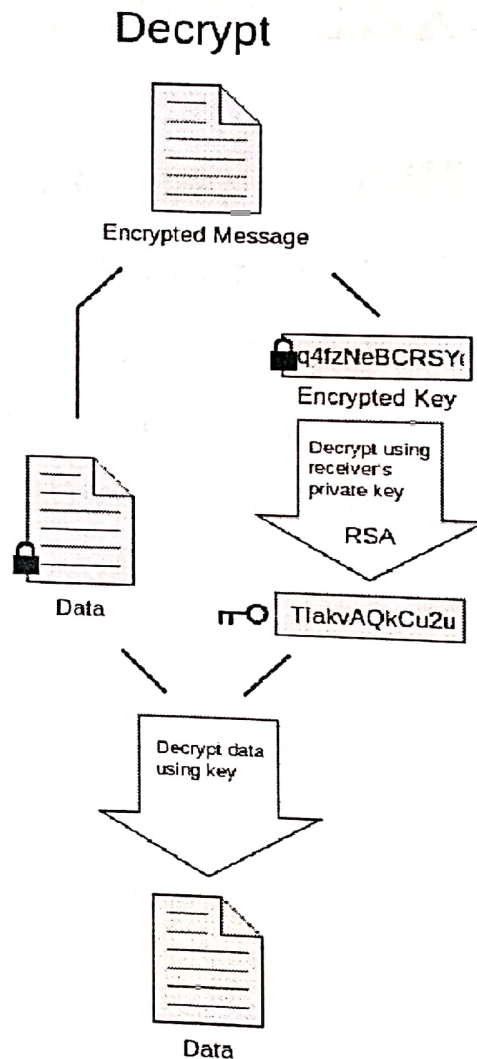
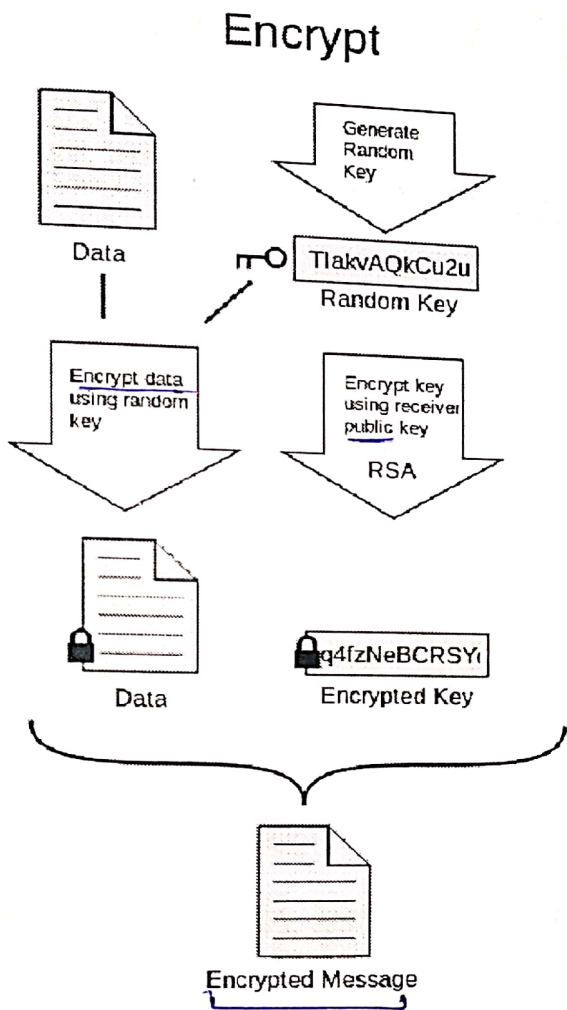
* PGP and other similar products follow the OpenPGP Standard (RFC 4880) for encrypting & decrypting data.



* Missing slide before this *

PGP

(مكتوبة في دفتر)



* شرح في
الدفتر

* missing slides here
ل (في دفتر)

* PGP :-

- * When using PGP, you will need to store:
 - your own Secret-Key (this will be stored encrypted with a passphrase).
 - your own public Key and the public Keys of your friends and associates (stored in the clear)
 - The PGP Software puts them in a file, called your Keyring.
 - your private Keys are in a file and stored encrypted with a Pass phrase.
 - The public Keys don't have to be protected.
 - The ~~A~~ Keyring also contains copies of other people's public Keys which are trusted by you.

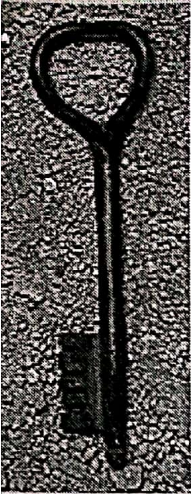
* PGP Tool :

- Gpg40 is one of the most popular PGP solutions for windows users and aims to integrate seamlessly with Outlook 2010 - 2016.
- All you have to do to encrypt your email is to select Sign Mail.
- Pros: Gpg40 offers simple handling for email, and integrates well with Outlook. For most windows users, it offers the easiest and most user-friendly PGP add-on out there.
- most people use PGP to send encrypted emails.
- the popularity of PGP has grown significantly. Huge numbers of people now use the standard to keep their private information private.

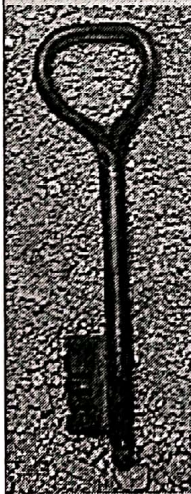
Example: PGP vs. SSL vs. IPsec

- ◆ SSL sits at the transport layer, "above" TCP
 - works only with TCP, doesn't work with UDP
 - Packet stream authenticated/encrypted
 - End-to-end security, best for connection-oriented sessions (e.g., http traffic)
 - Ports
 - User does not need to be involved
 - The OS does not have to change, but applications do if they want to communicate securely

Example: PGP vs. SSL vs. IPsec



- ◆ IPsec sits at the network layer
 - Individual packets authenticated/encrypted
 - End-to-end or hop-by-hop security
 - Best for connectionless channels
 - Need to modify OS
 - All applications are “protected” by default, without requiring any change to applications or actions on behalf of users
 - Only authenticates hosts, not users
 - User completely unaware that IPsec is running



SSL/TLS

Brief history...

- ◆ SSLv2 deployed in Netscape 1.1 (1995)
- ◆ Modified version of SSLv3 standardized as TLS

Broad overview

- ◆ SSL runs on top of TCP
 - Provides an API similar to that of TCP
↳ Application Programming Interface
- ◆ Technically, SSL runs in the application layer
 - Advantage: does not require changes to TCP
- ◆ From the programmer's point of view, it is in the transport layer
 - Same API as for TCP
 - Runs only with TCP, not UDP
- ◆ Primarily used for HTTP traffic



SSL overview

◆ Three phases (Stages)

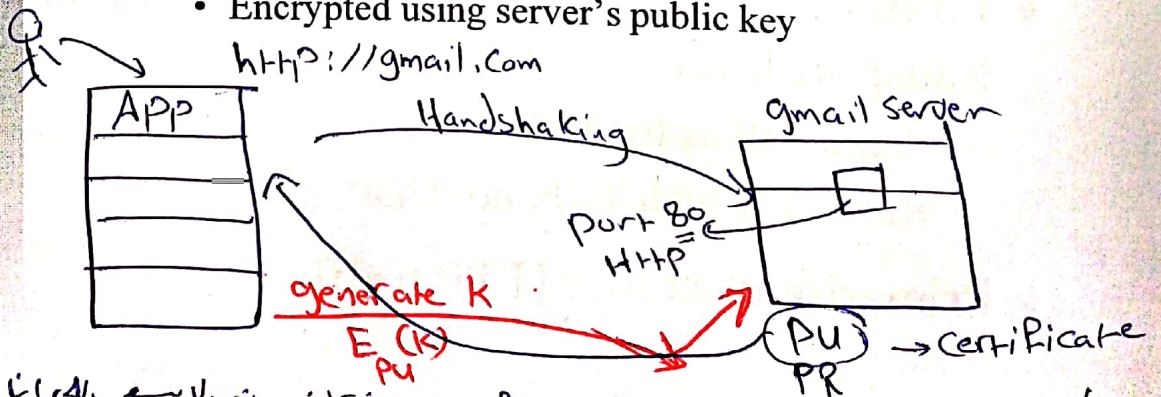
- Handshake
- Key derivation
- Data transfer



Handshake phase

◆ Client:

- ① Establishes TCP connection with server;
- ② Verifies server's identity
 - Obtains server's public key and certificate; verifies certificate
- ③ Sends server a master secret key K
 - Encrypted using server's public key



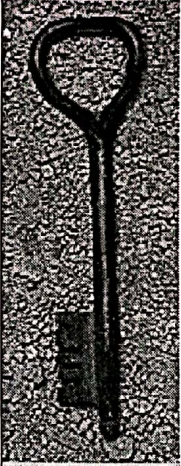
✗ لا يمكن التأكد من الهوية Certificate لا يمكن التأكد من الهوية
 website is insecure! gmail.com ←

Key derivation

- ◆ Client and server use K to establish four keys:
encryption and authentication, for each direction

Data transfer

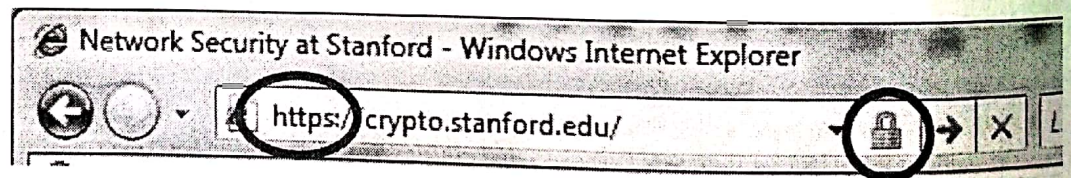
- ◆ SSL breaks data stream into *records*; appends a MAC to each record; and then encrypts the result
 - Mac-then-encrypt... → using 4 Keys.
- ◆ The MAC is computed over the record plus a sequence number
 - Prevents replay, re-ordering, or dropping packets



Note...

- ◆ As described, SSL only provides one-way authentication (server-to-client)
 - Not generally common for clients to have public keys
- ◆ Can do mutual authentication over SSL using, e.g., a password
 - SSL also allows for clients to have public keys

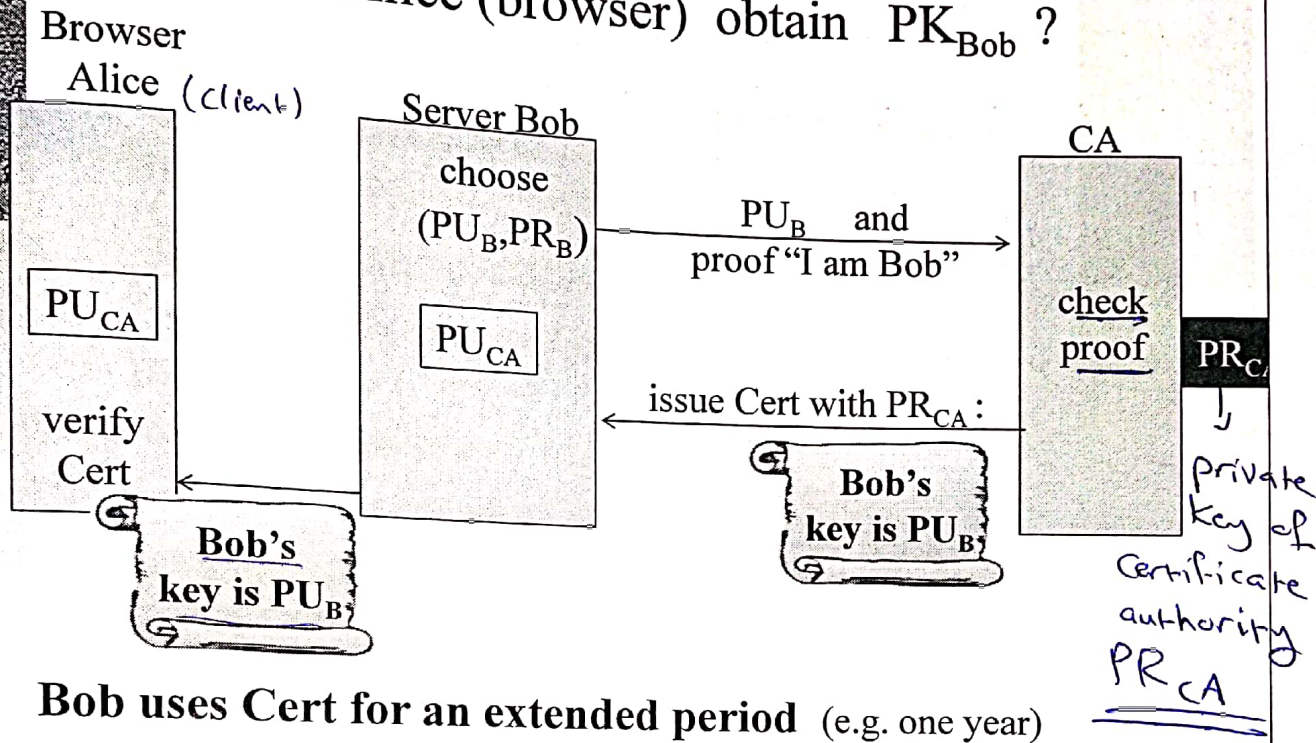
Public Keys for Servers in general.



HTTPS and the Lock Icon

Certificates

◆ How does Alice (browser) obtain PK_{Bob} ?



Certificates: example

◆ Important fields:

Certificate Signature Algorithm	
Issuer	
Validity	
Not Before	
Not After	
Subject	
Subject Public Key Info	
Subject Public Key Algorithm	
Subject's Public Key	
Extensions	
Field Value	
Modulus (1024 bits):	
ac 73 14 97 b4 10 a3 aa f4 c1 15 ed cf 92 f3 9a	
97 26 9a cf 1b e4 1b dc d2 c9 37 2f d2 e6 07 1d	
ad b2 3e f7 8c 2f fa a1 b7 9e e3 54 40 34 3f b9	
e2 1c 12 8a 30 6b 0c fa 30 6a 01 61 e9 7c b1 98	
2d 0d c6 38 03 b4 55 33 7f 10 40 45 c5 c3 e4 d6	
6b 9c 0d d0 8e 4f 39 0d 2b d2 e9 88 cb 2d 21 a3	
f1 84 61 3c 3a aa 80 18 27 e6 7e f7 b8 6a 0a 75	
e1 bb 14 72 95 cb 64 78 06 84 81 eb 7b 07 8d 49	

Certificate Viewer: *.gmail.com

General Details

This certificate has been verified for the following uses:
SSL Server Certificate

Issued To

Common Name (CN)	*.gmail.com
Organization (O)	Google Inc
Organizational Unit (OU)	<Not Part Of Certificate>
Serial Number	65:F8:33:2D:6B:CB:67:BC:AD:3A:80:A9:98:80:28:49

Issued By

Common Name (CN)	Thawte Premium Server CA
Organization (O)	Thawte Consulting cc
Organizational Unit (OU)	Certification Services Division

Validity

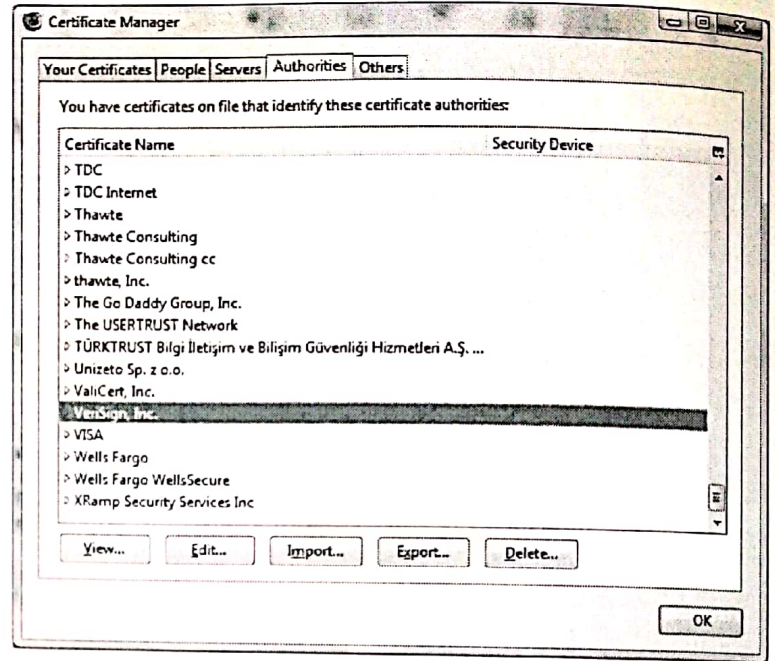
Issued On	9/25/2008
Expires On	9/25/2010

Fingerprints

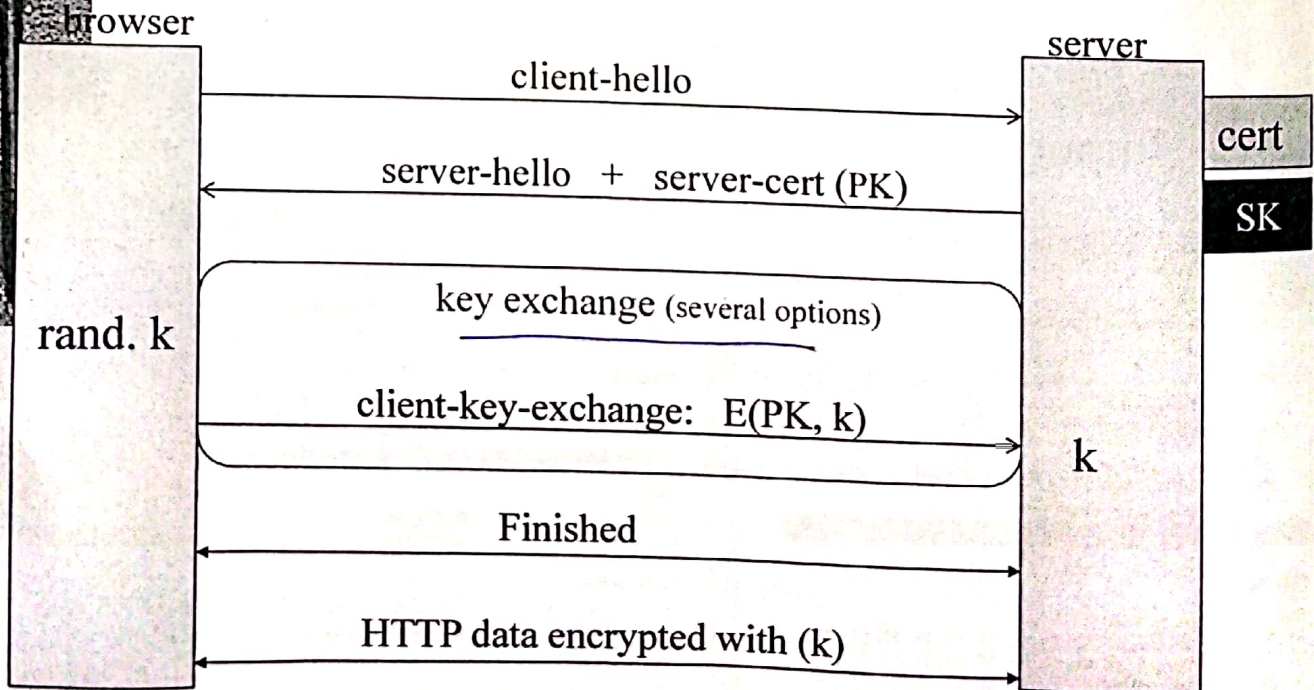
SHA1 Fingerprint	B7:A7:89:34:54:5D:C9:6F:41:FD:A9:3E:41:AF:2B:1D:13:C8:CC:AD
MDS Fingerprint	55:5F:09:17:24:03:F7:80:2B:B6:90:26:3B:0B:E3:3B

Certificate Authorities

Browsers accept certificates from a large number of CAs



Brief overview of SSL/TLS



Most common: server authentication only

Why is HTTPS not used for all web traffic?

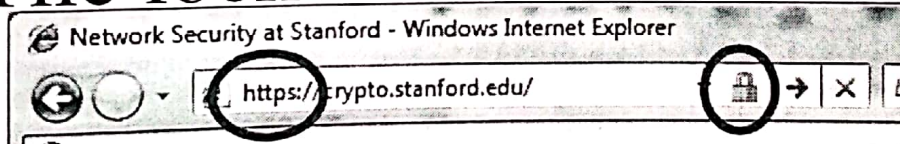
- 1. Slows down web servers

- 2. Breaks Internet caching

- ISPs cannot cache HTTPS traffic
- Results in increased traffic at web site

ای ایسی بڑی آمدہ ضار Web اور یجینی متار Web لانگ
پیر علی web proxy ← for caching

The lock icon: SSL indicator



Intended goal:

- Provide user with identity of page origin
- Indicate to user that page contents were not viewed or modified by a **network attacker**



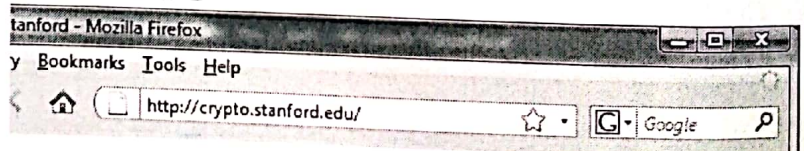
When is the (basic) lock icon displayed



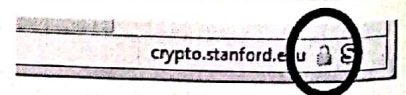
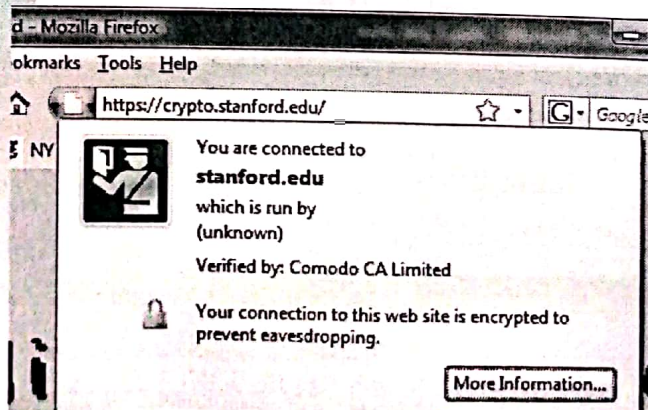
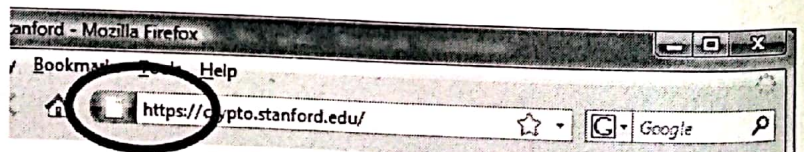
- All elements on the page fetched using HTTPS
 - For all elements:
 - HTTPS cert issued by a CA trusted by browser
 - HTTPS cert is valid (e.g. not expired)
 - CommonName in cert matches domain in URL
- certificate*

The lock UI: help users authenticate site

• Firefox 3: (no SSL)

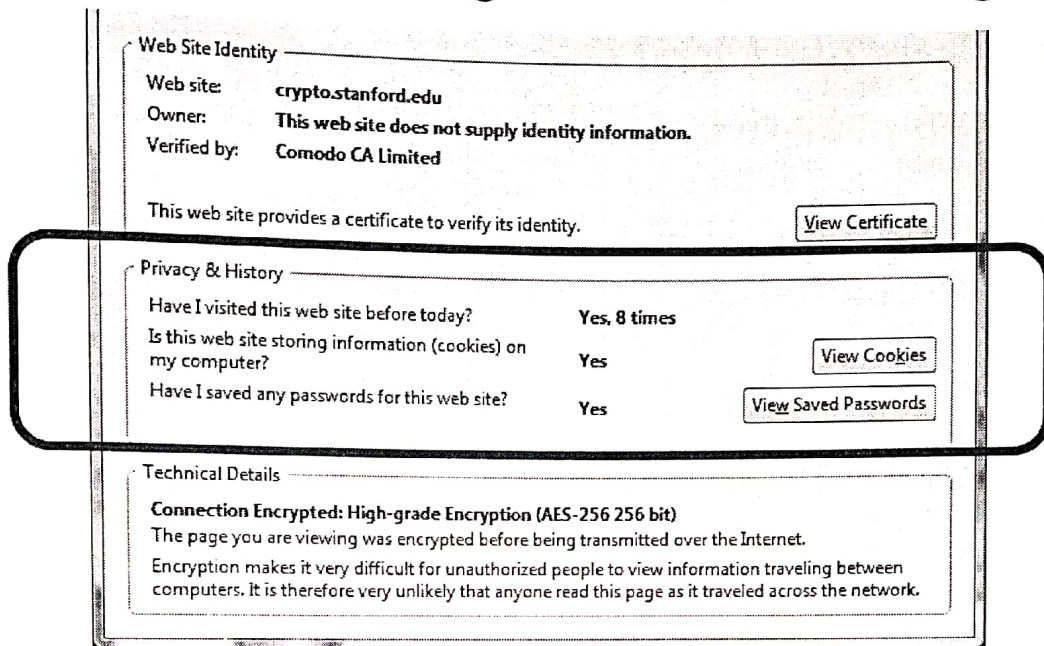


(SSL)



The lock UI: help users authenticate site

- ◆ Firefox 3: clicking on bottom lock icon gives

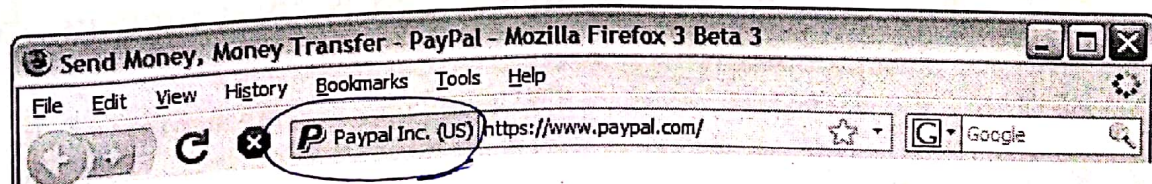


The screenshot shows the dropdown menu for the lock icon in Firefox 3. It is divided into three sections:

- Web Site Identity**
 - Web site: **crypto.stanford.edu**
 - Owner: **This web site does not supply identity information.**
 - Verified by: **Comodo CA Limited**
- Privacy & History** (highlighted with a red box)
 - Have I visited this web site before today? **Yes, 8 times**
 - Is this web site storing information (cookies) on my computer? **Yes** (with a **View Cookies** button)
 - Have I saved any passwords for this web site? **Yes** (with a **View Saved Passwords** button)
- Technical Details**
 - Connection Encrypted: High-grade Encryption (AES-256 256 bit)**
 - The page you are viewing was encrypted before being transmitted over the Internet.
 - Encryption makes it very difficult for unauthorized people to view information traveling between computers. It is therefore very unlikely that anyone read this page as it traveled across the network.

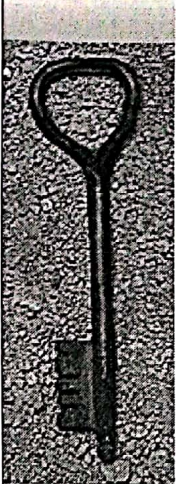
The lock UI: Extended Validation (EV) Certs

- Harder to obtain than regular certs
 - requires human lawyer at CA to approve cert request
 - more cost, complicated
- Designed for banks and large e-commerce sites



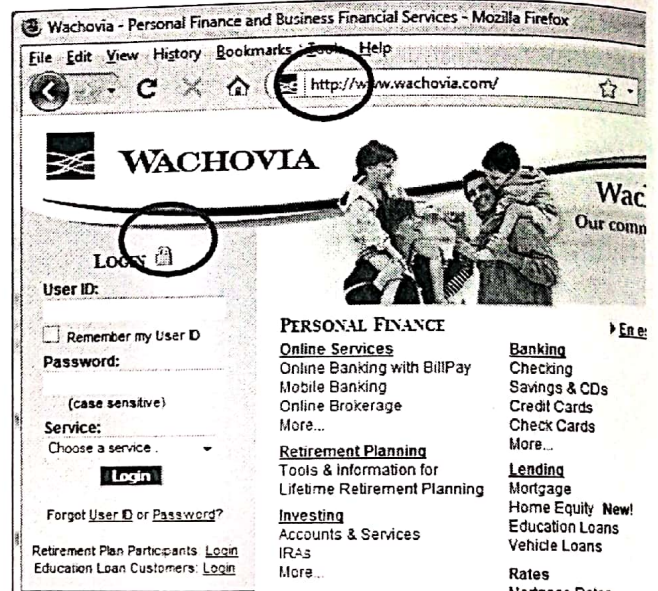
↳ extended validation.

HTTPS and login pages: incorrect version

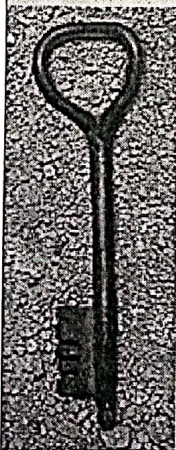


◆ Users often land on login page over HTTP:

- Type site's HTTP URL into address bar, or
- Google links to the HTTP page



Invalid certs



➤ Examples of invalid certificates:

- ① expired: current-date > date-in-cert
- ② CommonName in cert does not match domain in URL
- ③ unknown CA (e.g. self signed certs)
 - Small sites may not want to pay for cert

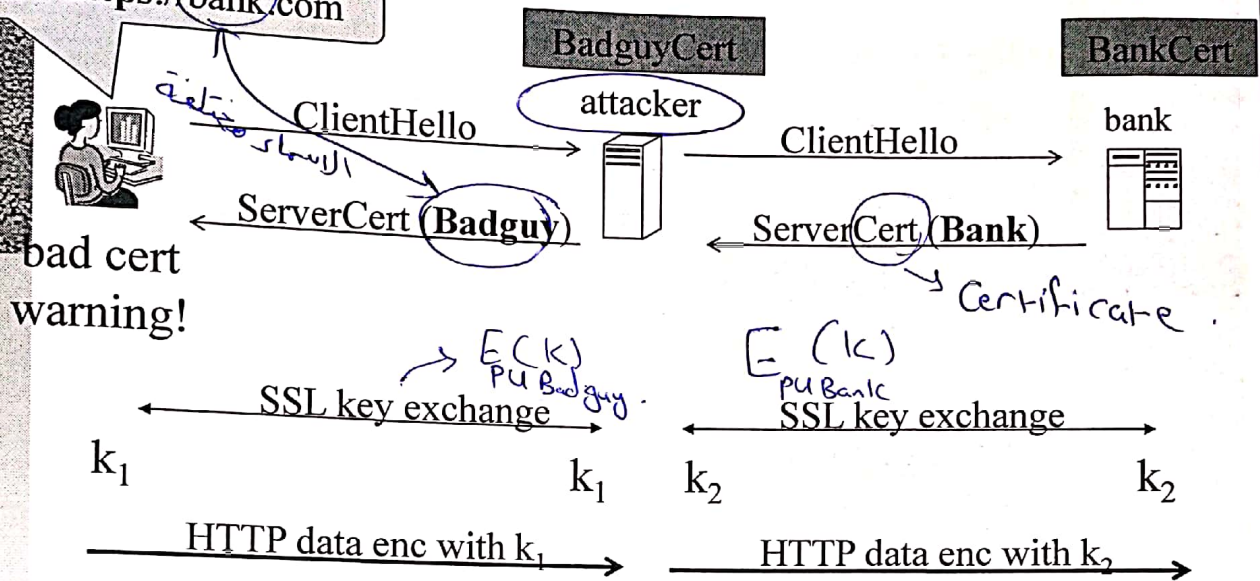
➤ Users often ignore warning:

- Is it a miss-configuration or an attack? User can't tell.
- ◆ Accepting invalid cert enables man-in-middle attacks

(see <http://crypto.stanford.edu/ssl-mitm>)

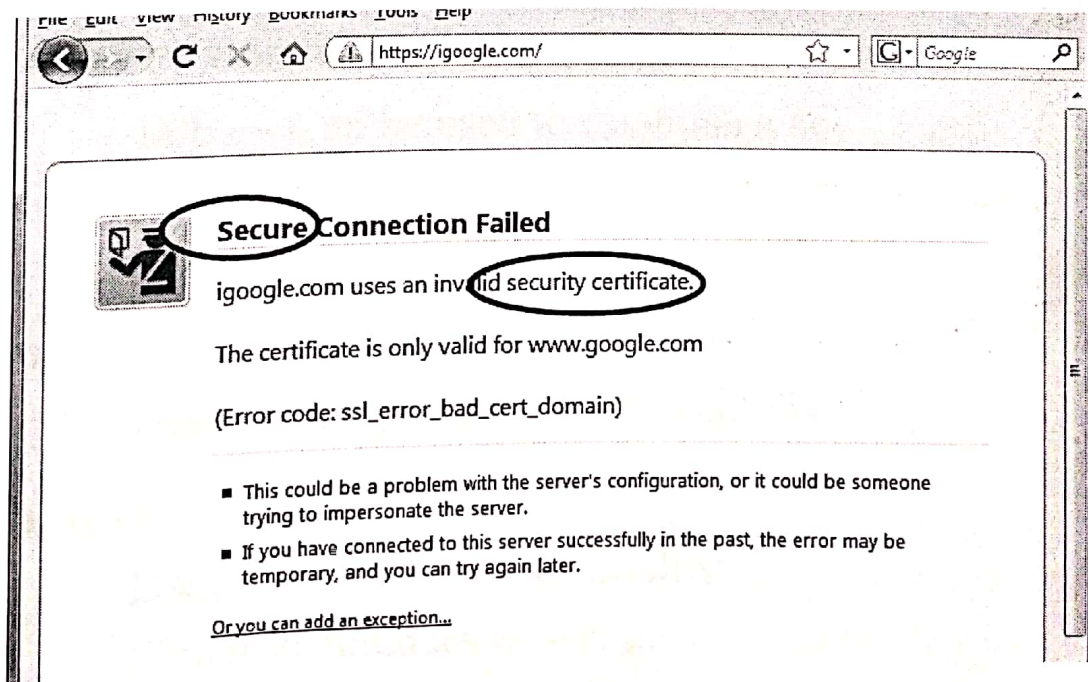
Man in the middle attack using invalid certs

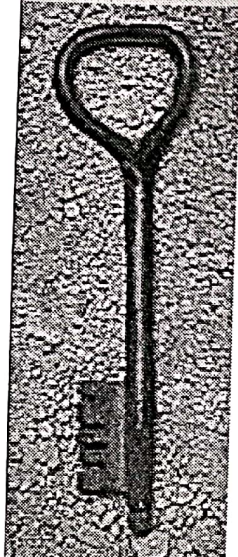
GET https://bank.com



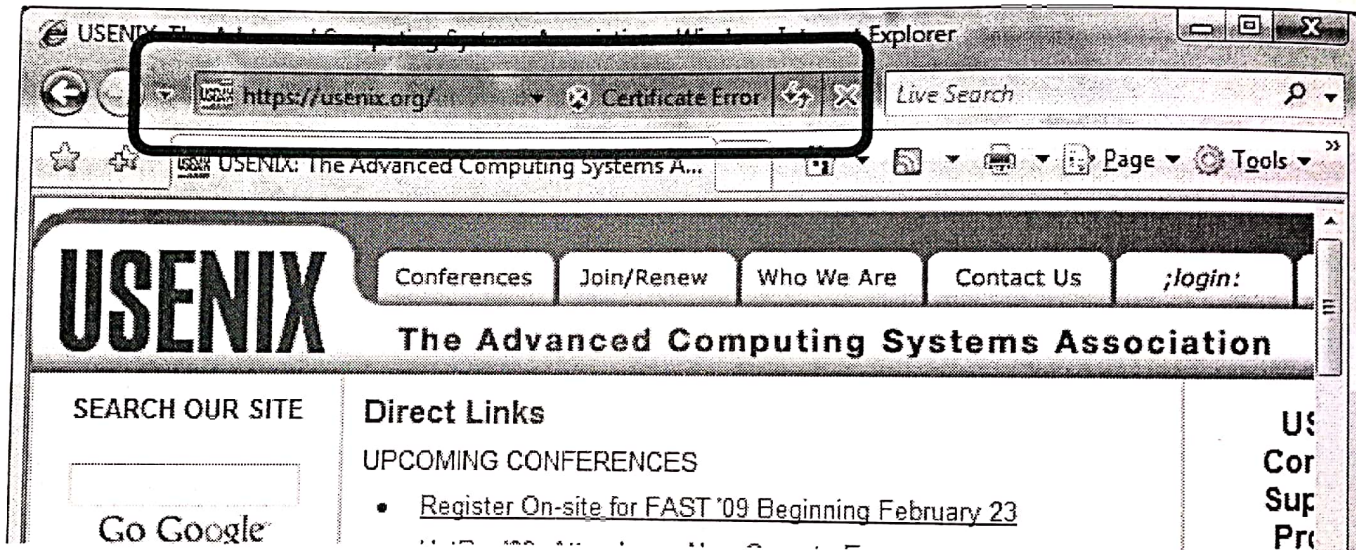
- ◆ Attacker proxies data between user and bank.
Sees all traffic and can modify data as will.

Firefox: Invalid cert dialog





IE: invalid cert URL bar



* SSH :-

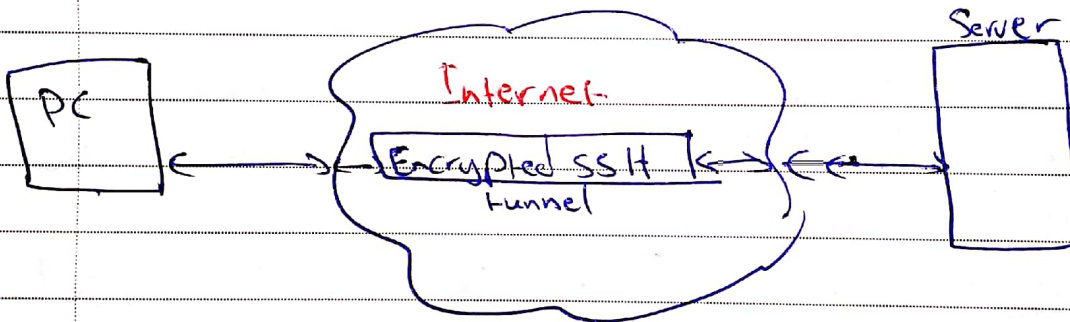
- * Secure Shell or SSH is a network protocol that allows data to be exchanged using a secure channel between two networked devices.
- * Very common on Unix based systems, it is used as a secure replacement of Telnet as it allows remote access to a computer through a secure shell.
- * A client connecting to a SSH server, will have shell access on the server, in a secure way.
- * SSH, by means of public keys can enforce authentication for both client & server.
- * Moreover it is also used to create tunnels, port forwarding and secure file transfer.
- * An SSH server, by default, listens on TCP port 22.

Notes:

- * Telnet - Not secure even if it requires password, because there is no encryption.
- * Port number is 16 bits
 - o - 1024 reserved port numbers

* SSH Tunnel:

- * An SSH tunnel is an encrypted tunnel created through an SSH protocol connection.
- * SSH tunnels maybe used to tunnel unencrypted traffic over a network through an encrypted channel.
- * SSH allows one to tunnel any protocol within a secure channel.
- * It can be used to add encryption to legacy applications.
- * you can do so for instant messaging protocols, mount remote hard drives and so on.

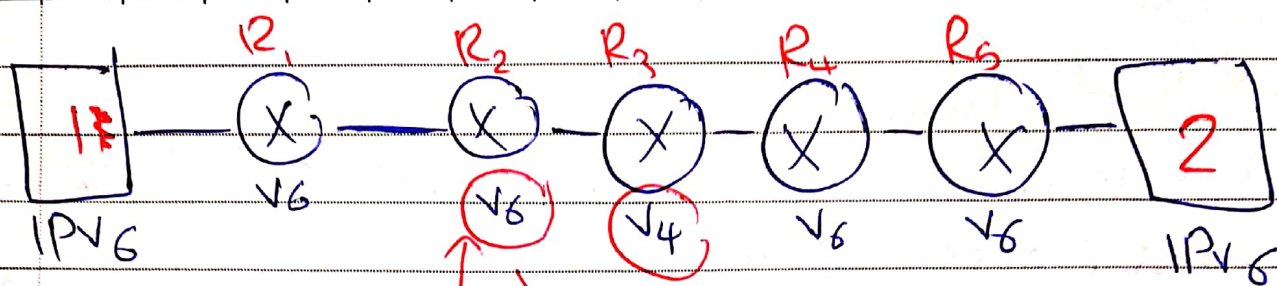


Notes:

3 types of SSH tunnel:

- ↳ local forwarding
- ↳ Remote forwarding
- ↳ Dynamic port forwarding.

* example in lecture 2 / week 12 : at minute 21:00



knows that next router doesn't support V6 → by hello packets

→ So R2 tunnels (encapsulates) the packet as if it's V4 R2 → src IP, R4 → dst IP when the packet passes R3, Now R4 remove the tunneling & continue sending it to the destination.

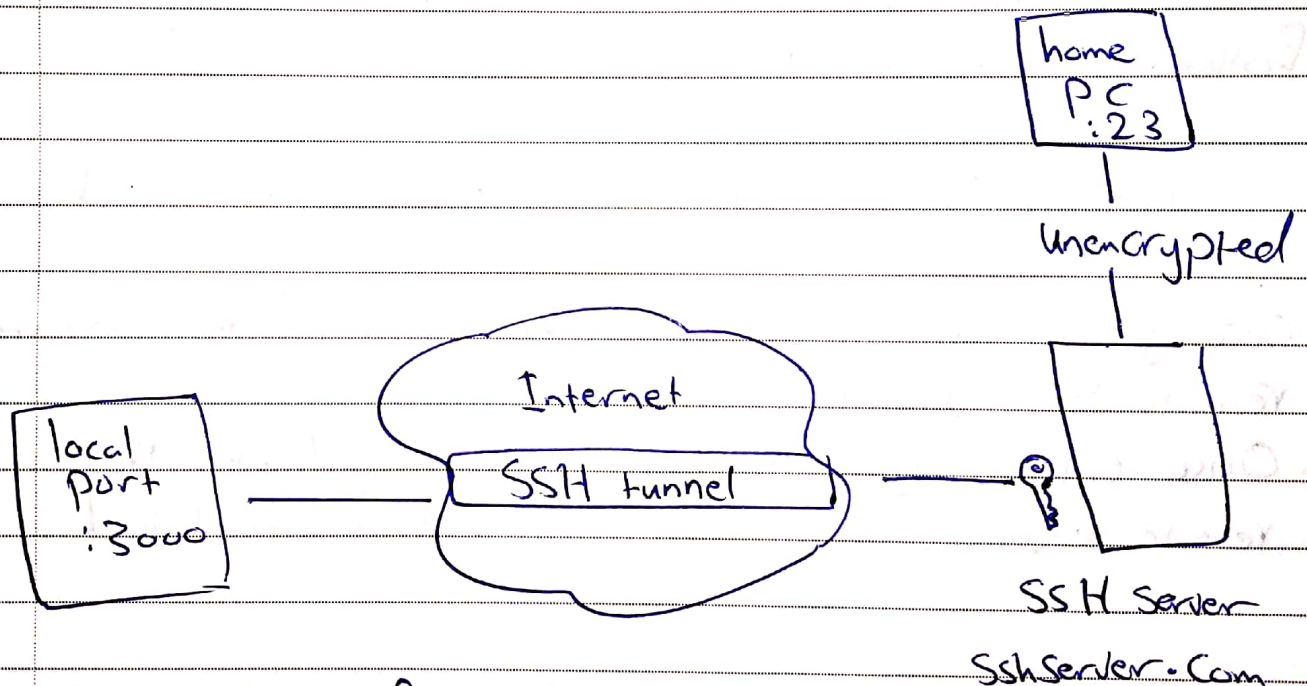
- * To Create an SSH Tunnel, an SSH Client is configured to forward a specified local port to a port on the remote machine.
- * Traffic to the local port (SSH Client) is forwarded to the remote host (SSH Server).
- * The remote host will then forward this traffic to the intended target host.
- * The traffic between SSH Client and Server will be encrypted.
- * SSH Tunnels provide a means to bypass firewalls that prohibit certain Internet services provided that outgoing connections are allowed.
- * Corporate policies and filters can be bypassed by using SSH Tunnels.

* SSH Tunnel :

→ local forwarding

```
ssh -L 3000:homepc:23 Bob@sshserver.com
```

With this command, all the traffic sent to local host's port 3000 will be forwarded to remote host on port 23 through the tunnel.



- L : Local Forwarding

- R : Remote Forwarding

- D : Dynamic Forwarding

* SSH Example:

* Let's say you have a MySQL database server running on machine `dbool.host` on an internal (private) network, on port- 3306, which is accessible from the machine `pubool.host`, and you want to connect using your local machine MySQL client to the database server

→ To do so, you can forward the connection using the following command:

```
ssh -L 3336 :dbool.host :3306 user@pubool.host
```

* Once you run the command, you'll be prompted to enter the remote SSH user password

* Once entered, you'll be ~~logged~~ logged into the remote server, and the SSH tunnel will be established.



Set up SSH Tunneling in Windows

PuTTY Configuration

Category:

- Session
- Logging
- Terminal
 - Keyboard
 - Bell
 - Features
- Window
 - Appearance
 - Behaviour
 - Translation
 - Selection
 - Colours
- Connection
 - Data
 - Proxy
 - Telnet
 - Rlogin
 - SSH**
 - Serial

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address)	Port
192.168.122.1	22

Connection type:

Raw Telnet Rlogin SSH Serial

Load, save or delete a stored session

Saved Sessions

--

Default Settings

Load Save Delete

Close window on exit:

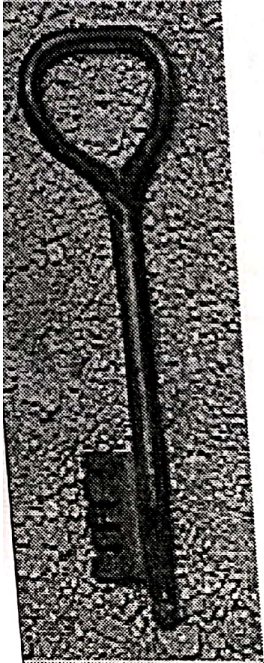
Always Never Only on clean exit

About Help Open Cancel

SSH Server and port number

- ◆ Under the Connection menu, expand SSH and select Tunnels.
 - Check the Local radio button to setup local, Remote for remote, and Dynamic for dynamic port forwarding.

- ◆ When setting up local forwarding, enter the local forwarding port in the Source Port field and in Destination enter the destination host and IP,
 - for example, localhost:5901 if the server on the SSH server.
 - Or, db-Server:5901 if the server db-Server is different from the SSH server.



IPsec

Security at the IP layer
(Network layer)

Overview

gives security to all above it

- ◆ IPsec can provide security between any two network-layer entities
 - host-host, host-router, router-router
- ◆ Used widely to establish VPNs → Virtual private Network
- ◆ IPsec encrypts and/or authenticates network-layer traffic, and encapsulates it within a standard IP packet for routing over the Internet

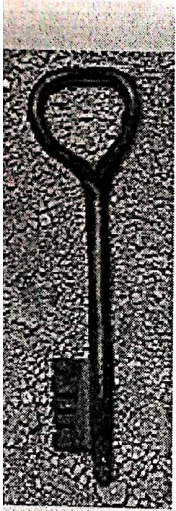
* user has No choice of Security.

Overview

- ◆ IPsec consists of two components
 - ① – IKE --- Can be used to establish a key
 - ② – AH/ESP --- Used to send data once a key is established (whether using IKE or out-of-band) (data exchange)
- ◆ AH (only Authentication)
 - Data integrity, but no confidentiality
- ◆ ESP (Authentication + Confidentiality (Encryption))
 - Data integrity + confidentiality
 - (Other differences as well)

optional

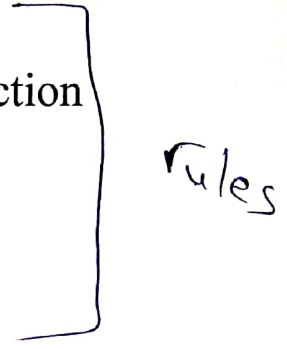
like firewalls



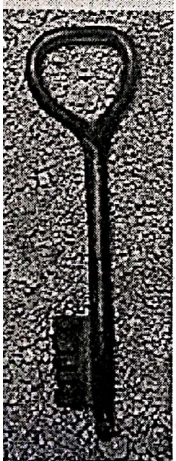
Security policy database

التسريح في دفتر

- ◆ Nodes maintain a table specifying what is required for each incoming packet
 - Drop
 - Forward/accept without IPsec protection
 - Require IPsec protection
 - Auth only
 - Enc only
 - Both



- ◆ As with firewalls, decisions can be based on any information in the packet



Security associations (SAs)

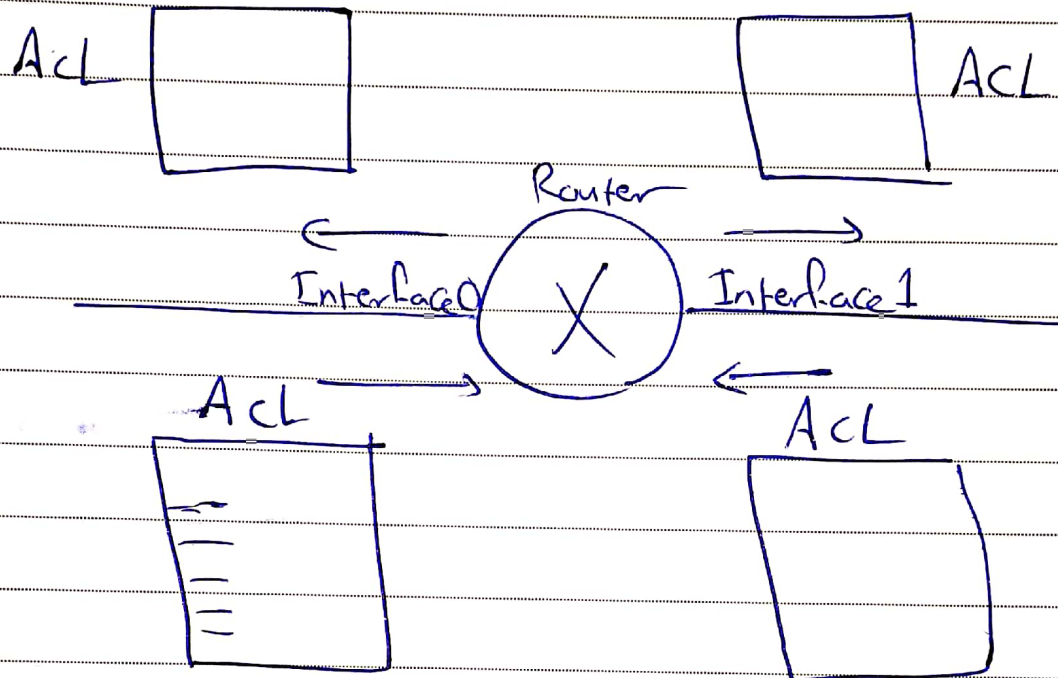
One way → From SRC to dst or From dst to SRC

- ◆ When a node receives a packet, needs to know who it is from
 - May be receiving IPsec traffic from multiple senders at the same time -- possibly even with the same IP address
- ◆ An SA defines a network-layer unidirectional logical connection
 - For bidirectional communication, need two SAs
- ◆ The IPsec header indicates which security association to use

From source to destination

يعني هاد ال SRC - dst طبقت عليه هاي ال security policy

* Security Policy database



* ACL: Access control list
rules of Deny, permit.

When the packet received, it reads the header then either accept or deny.

→ Any Router with ACL is Firewall.



Security associations (SAs)

◆ A tremendous amount of information is kept in the SADB, and we can only touch on a few of them:

- AH: authentication algorithm
- AH: authentication secret
- ESP: encryption algorithm
- ESP: encryption secret key
- ESP: authentication enabled yes/no → (optional)
- Many key-exchange parameters
- Routing restrictions
- IP filtering policy

→ Security Association DataBase



Firewalls...

Filtering data depending on set of rules.

- ◆ Potential problem if upper-layer header data is used for decision-making; this information will be encrypted when using IPsec
- ◆ Arguments pro and con as to whether this data should be encrypted or not:
 - Pro: This data shouldn't be divulged; get rid of firewalls
 - Con: administrators will likely keep firewalls and turn off encryption...

AH vs. ESP

- ◆ Two header types...
- ◆ Authentication header (AH)
 - Provides integrity only
- ◆ Encapsulating security payload (ESP)
 - Provides encryption + integrity
- ◆ Both provide cryptographic protection of everything beyond the IP headers
 - AH additionally provides integrity protection of some fields of the IP header

IP header

الجزء الوحيدة لـ AH مقارنة بالـ ESP

two modes for AH - ESP:

① Transport vs. ② tunnel mode

- ◆ Transport mode: original IP header not touched; IPsec information added between IP header and packet body

– IP header | (IPsec) | [packet]
protected

- Most logical when IPsec used end-to-end

Transport vs. tunnel mode

Router-to-Router / Firewall-to-Firewall

- ◆ Tunnel mode: keep original IP packet intact but protect it; add new header information outside

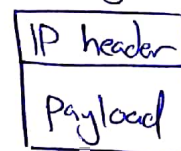
– New IP header | IPsec | [old IP header | packet]

↓
to encapsulate the data.

encrypted
authenticated

- Can be used when IPsec is applied at intermediate point along path (e.g., for firewall-to-firewall traffic)
 - Treat the link as a secure tunnel
- Results in slightly longer packet

Datagram



More on AH

- ◆ AH provides integrity protection on header

– But some fields change *en route*!

- ◆ Immutable fields included in the integrity check

- ◆ Mutable but predictable fields are also included in the integrity check

– The *final* value of the field is used

یعنی ہر وقت ہونے والا

Final Value

مقام کی طرف

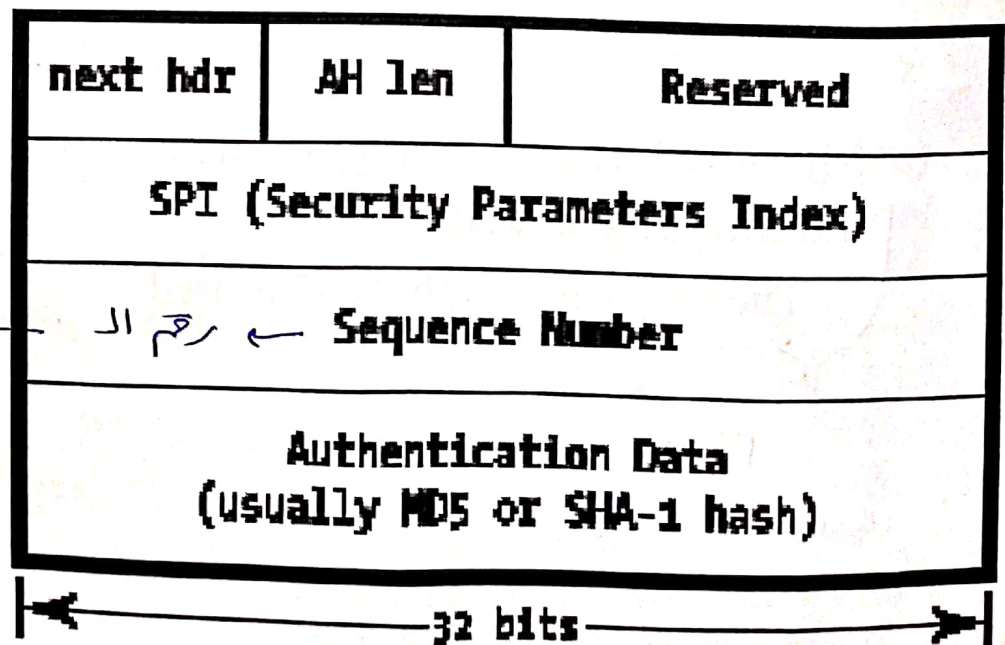
More on AH vs. ESP

→ doesn't protect header

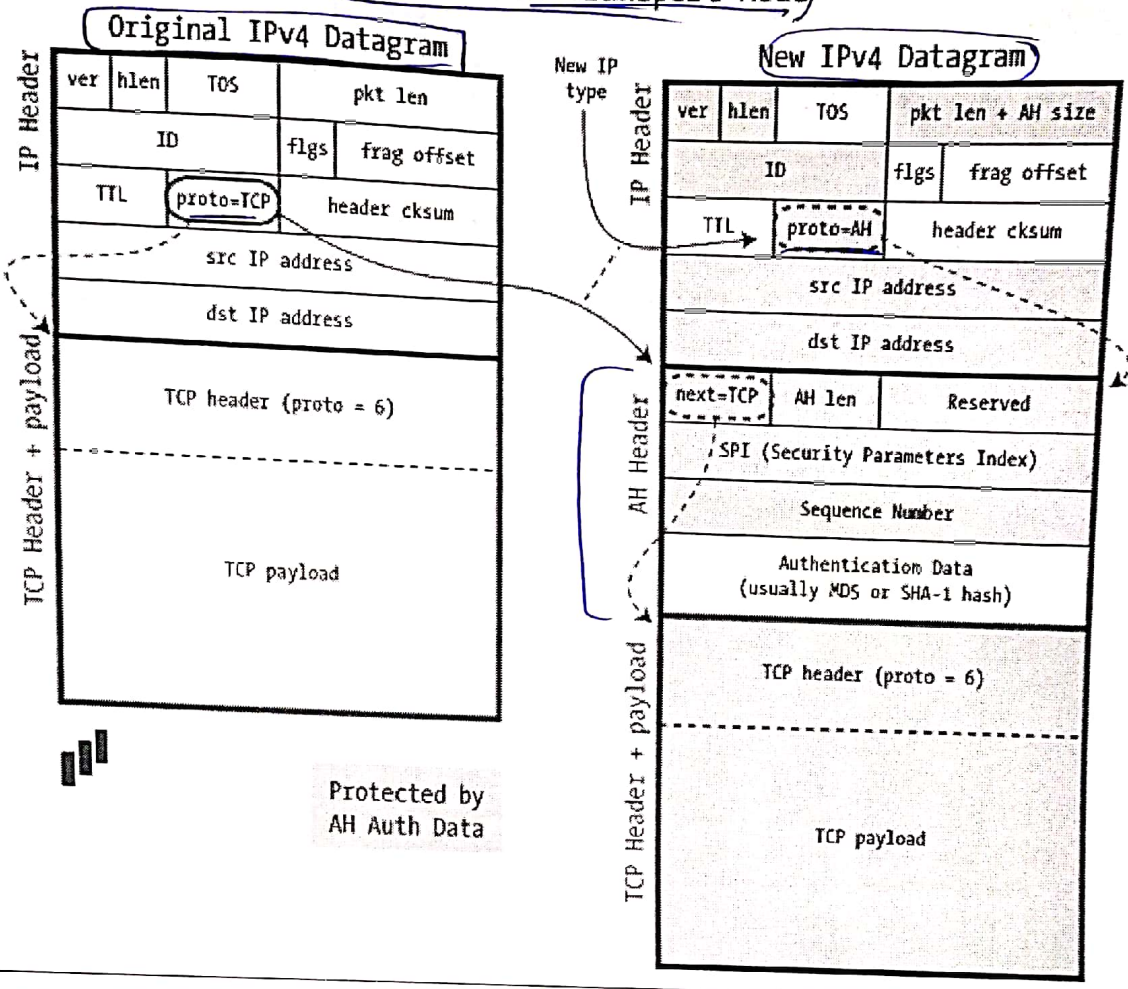
- ◆ ESP can already provide encryption and/or authentication
- ◆ So why do we need AH?
 - AH also protects the IP header
 - Export restrictions
 - Firewalls need some high-level data to be unencrypted

AH Header

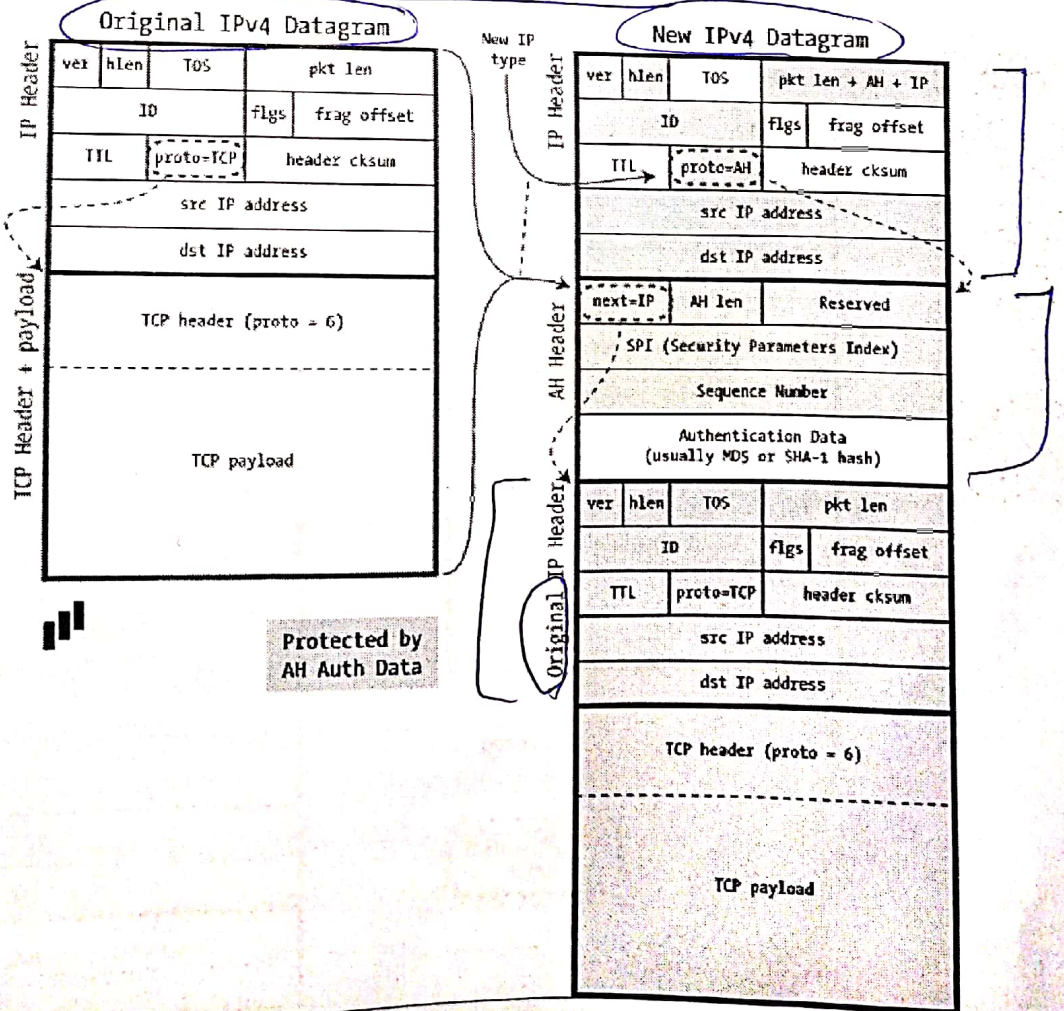
IPSec AH Header



IPSec in AH Transport Mode

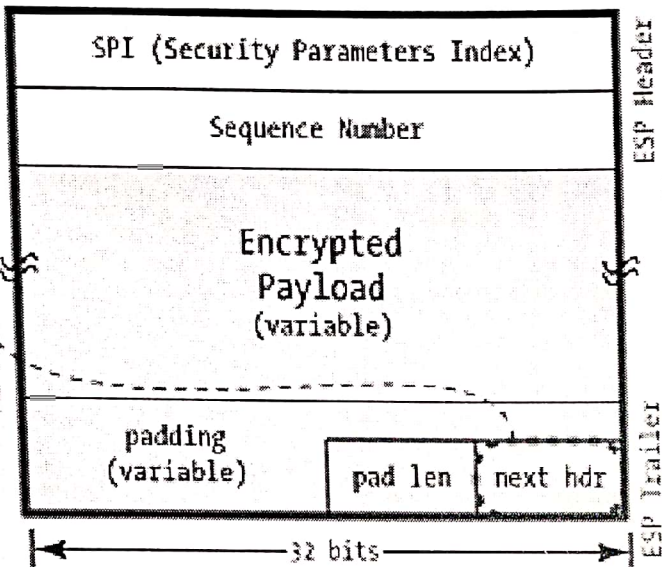


IPSec in AH Tunnel Mode

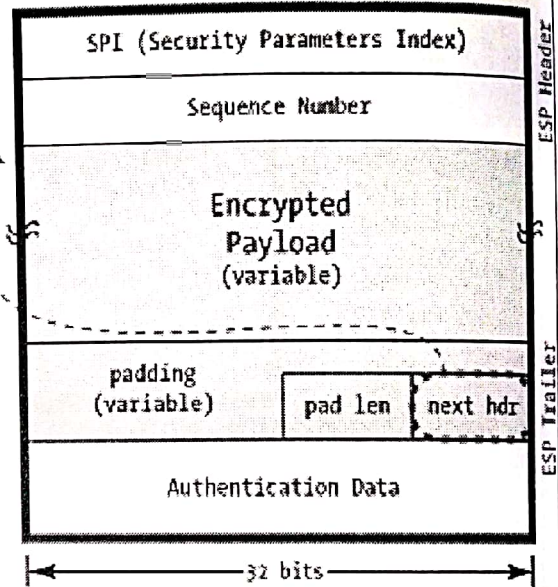


ESP Header

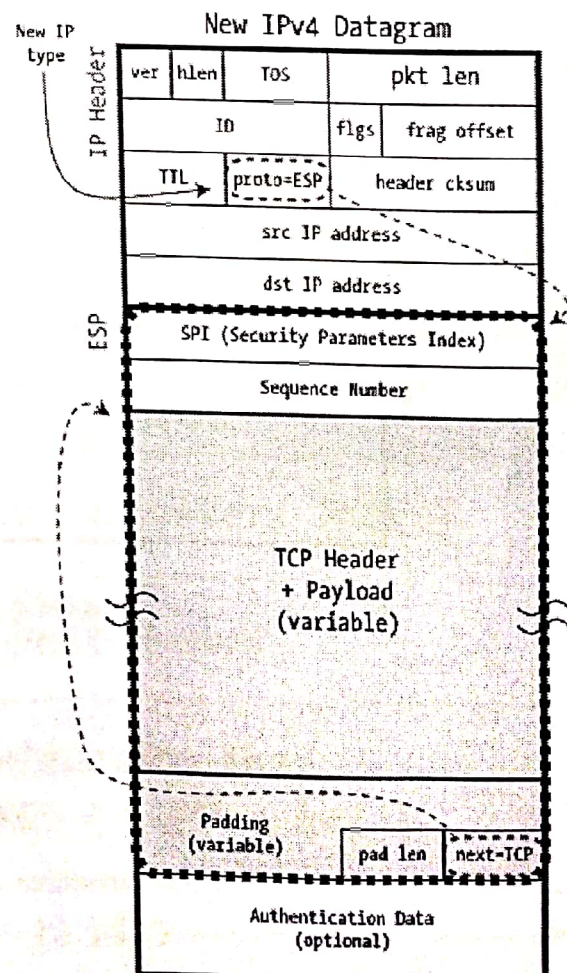
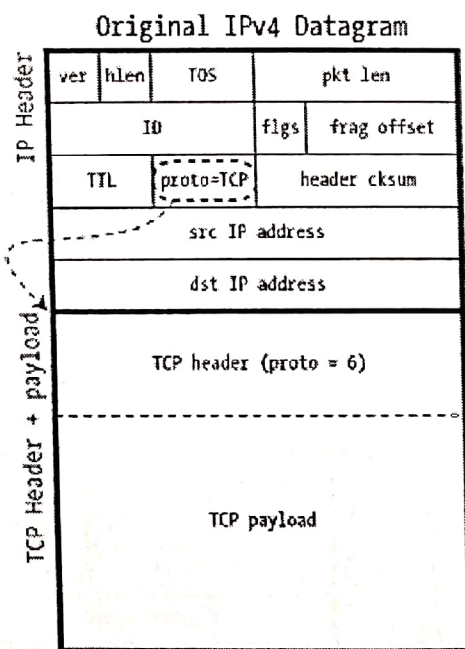
ESP w/o Authentication



ESP with Authentication



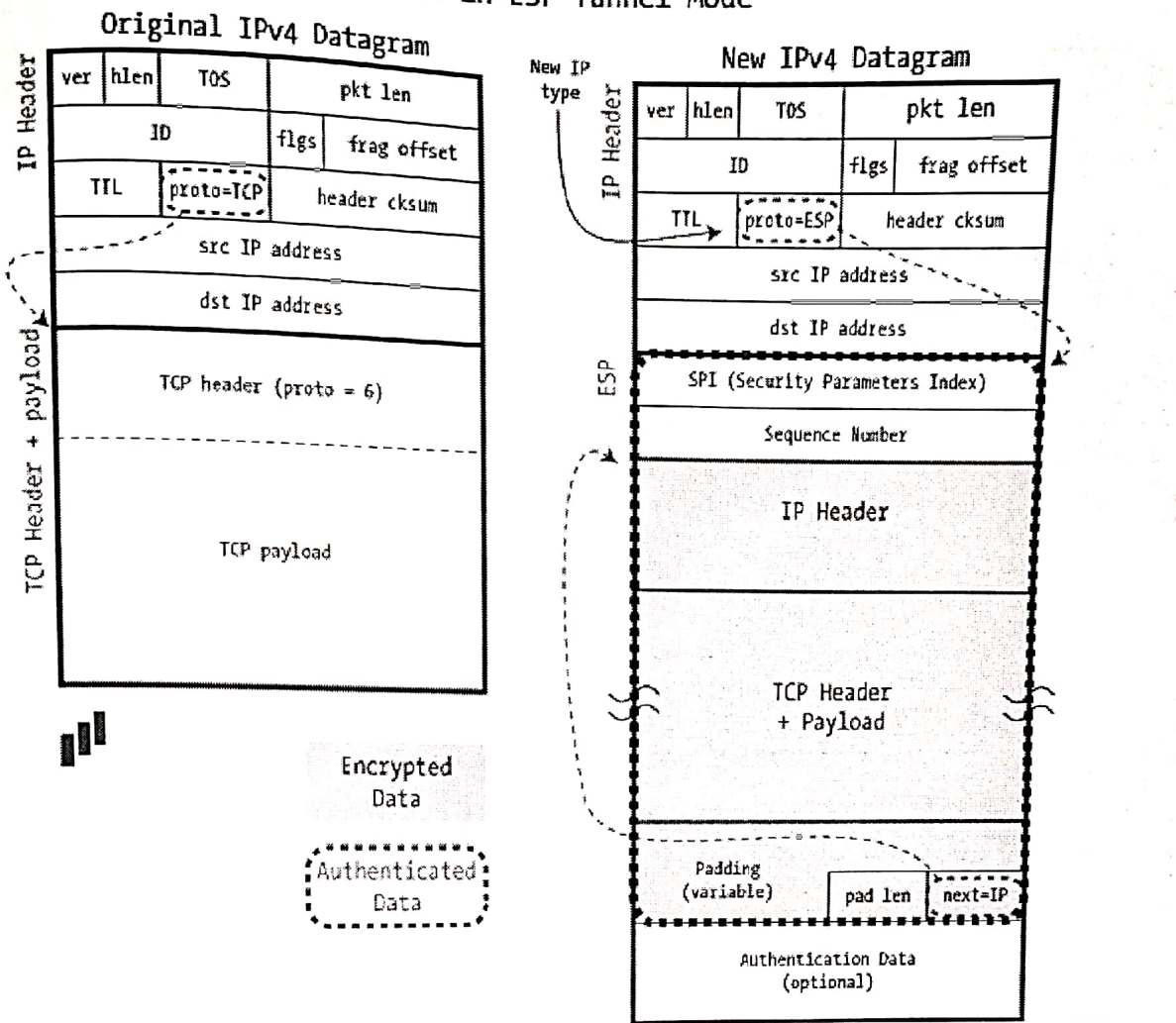
IPSec in ESP Transport Mode



Encrypted Data

Authenticated Data

IPSec in ESP Tunnel Mode



The future of AH?

- ◆ In the long run, it seems that AH will become obsolete
 - Better to encrypt everything anyway
 - No real need for AH
 - Certain performance disadvantages
 - AH is complex...



IPsec: IKE



Overview of IKE

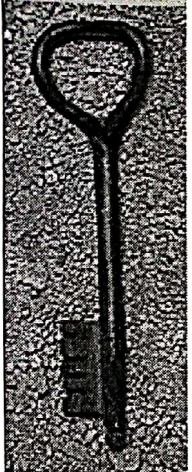
- ◆ IKE provides mutual authentication, establishes shared key, and creates SA
- ◆ Assumes a long-term shared key, and uses this to establish a session key (as well as to provide authentication)
- ◆ Supported key types
 - Public signature keys
 - Public encryption keys
 - Symmetric keys

IKE phases

- ◆ Phase 1: long-term keys ^① used to derive a session key (and provide authentication)
- ◆ Phase 2: session key ^② used to derive SAs
- ◆ Why...?
 - In theory, can run phase 1 once, followed by multiple executions of phase 2
 - E.g., different flows between same endpoints
 - Why not used same key for each? Is there any secure way to do this?
 - In practice, this anyway rarely happens

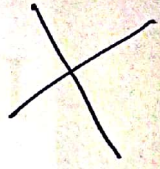
Key types

- ◆ Why are there two PK options?
 - Signature-based option
 - Efficiency (can start protocol knowing only your own public key, then get other side's key from their certificate)
 - Legal reasons/export control
 - Encryption-based option
 - Can be used to provide anonymity in both directions
- ◆ Adds tremendously to the complexity of implementation

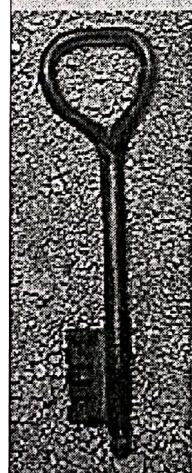


Anonymity

التي مجموعة سلاسل
كالحاها



- ◆ Protocols can be designed so that identities of the parties are hidden from eavesdroppers
 - Even while providing authentication!
- ◆ Can also protect anonymity of one side against active attacks
 - Whom to protect?
 - Initiator: since responder's identity is generally known...
 - Responder: since otherwise it is easy to get anyone's identity



Phase 1 session keys

- ◆ Two session keys are defined in phase 1
 - One each for encryption/authentication
- ◆ These keys are used to protect the final phase 1 messages as well as all phase 2 messages
- ◆ These keys are derived from the DH key using hashing
 - Details in the book...

Diffie Helman

IKE phase 1

- ◆ Aggressive mode
 - 3 messages

Both uses Diffie Helman

- ◆ Main mode
 - 6 messages

- Additional features:

- Anonymity (Identity protection)
- Negotiation of crypto parameters

Aggressive mode

3 messages

- ① Alice sends g^a , "Alice", crypto algorithms } at first
- Note that choices are restricted by this message } Not encrypted
- ② Bob sends g^b , choice of crypto algorithm, "proof" that he is really Bob } Identity
- ③ Alice sends "proof" that she is Alice



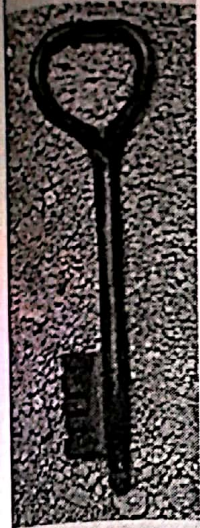
Main mode

- ◆ Negotiate crypto algorithms (2 rounds)
- ◆ Alice and Bob do regular Diffie-Hellman key exchange (2 rounds)
- ◆ Alice sends encryption of "Alice" plus a proof that she is Alice, using long-term secret keys plus [keys derived from] g^{ab}
- ◆ Bob does similarly...



Crypto parameters...

- ◆ Choice of:
 - Encryption method (DES, 3DES, ...)
 - Hash function (MD5, SHA-1, ...)
 - Authentication method (e.g., key type, etc.)
 - Diffie-Hellman group (e.g., (g, p), etc.)
- ◆ A complete set of protocols (a security *suite*) must be specified



Negotiating parameters

- ◆ Many protocols allow parties to negotiate cryptographic algorithms and parameters
 - Allows users to migrate to stronger crypto; increases inter-operability (somewhat)
- ◆ But, opens up a potential attack if not authenticated somehow...
- ◆ Also makes for more complicated implementations



“Proofs of identity”

- ◆ Depend on which type of long-term shared key is being used
- ◆ Similar (in spirit) to the authentication protocols discussed in class
 - Details in book...



Anonymity

By: Dr. Ramzi Saifan

No.

* Anonymity :-

عملية إخفاء (اختفاء IP) بقدر ما

إتاحة قسرين وبتوافق

عملية ال Encryption فقط و data و payload من عملية
Anonymity و

Anonymity types:

① Browsing Anonymity

② Anonymous from Intermediate devices (All)



Browsing Anonymously

- ◆ If you want to be anonymous, then you need to find a *service* that provides anonymity.
 - You are exposing all data you see to that person/company as they can sniff the data.
 - Therefore, use extreme caution and do not use any systems that you do not own, or
 - that you do not know what information they capture.

* Browsing Anonymously :

- From browser settings for proxy
- Through web proxy

بکلی ال Traffic قطع کی ال Proxy ال ہو سکتا ال Gateway ال
ال Proxy ال کن کیوں جو ال LAN ال ہو ال

از ال Proxy ال بعد ال ال IP ← as Cache
از ال بعد ال ال ال IP ← For Anonymity

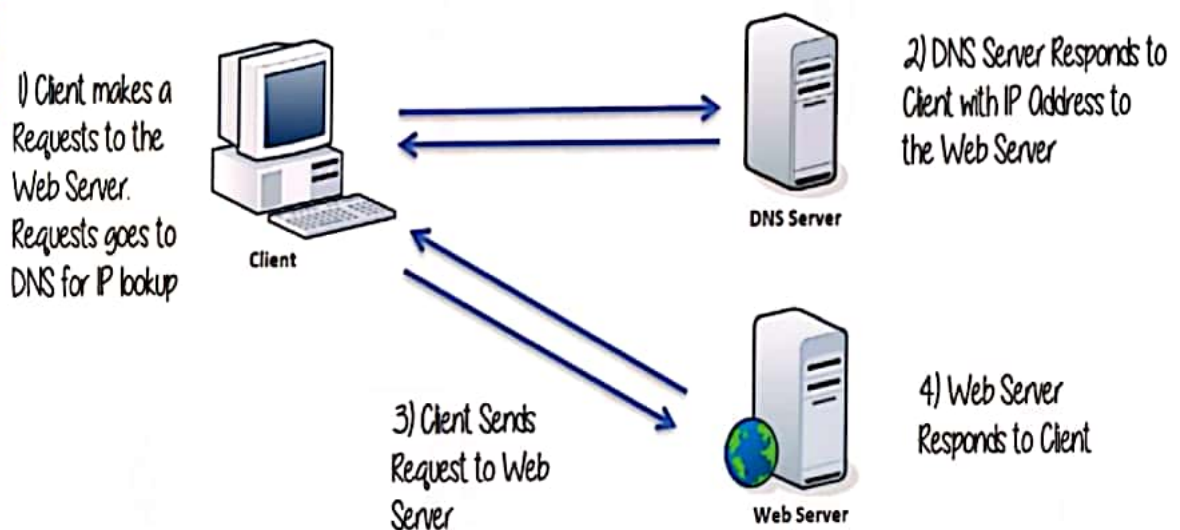


HTTP Proxies

- ◆ Using proxies is basically asking another system to do something on your behalf.
- ◆ Instead of sending a request directly to a web server, you would send the request to a proxy server first.
- ◆ The proxy server works on your behalf to request the web page, and subsequently sends it back to you.
- ◆ This causes the web server to see the proxy servers address, not yours.

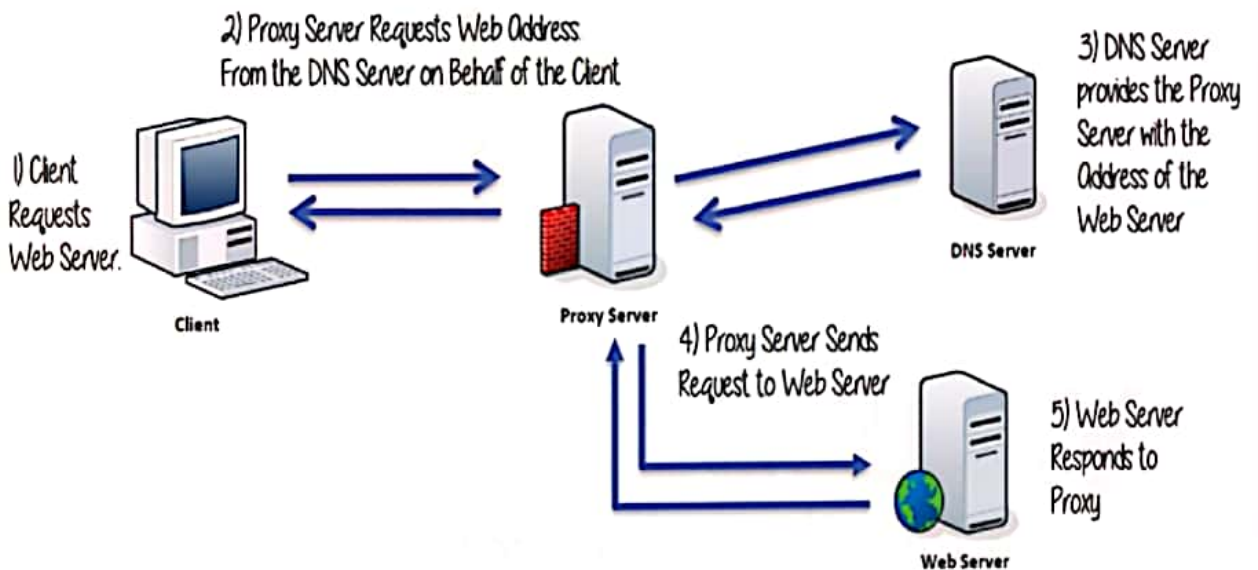


Normal flow of web requests





Flow of web requests proxy.



Types of Web Proxies

- ◆ Many times, these proxy servers can be misconfigured web servers that enabled the proxy services.
- ◆ There are two general types of proxies:
 - Require you to change your web browser settings
 - you have to know the services the proxy handles and the corresponding ports: HTTP, SSL/HTTPS, FTP, Gopher, or Socks are the most common types.
 - Other that are used through their web pages.



Proxies through their web pages

The checkbox under the HTTP proxy settings sets the same settings for all other proxy types.

Configure Proxies to Access the Internet

No proxy
 Auto-detect proxy settings for this network
 Use system proxy settings
 Manual proxy configuration:

HTTP Proxy: Port: :
 Use this proxy server for all protocols

SSL Proxy: Port: :
FTP Proxy: Port: :
SOCKS Host: Port: :
 SOCKS v4 SOCKS v5 Remote DNS

No Proxy for:

Example: .mozilla.org, .net.nz, 192.168.1.0/24

Automatic proxy configuration URL:

Do not prompt for authentication if password is saved



Your IP

- ◆ <http://www.whatsmyip.org>.
- ◆ <http://www.checkip.org/>
- ◆ <http://whatismyipaddress.com/>

Gives you your public IP



My IP without proxy

The first time we visit the site without a proxy, we can see that the box shows our current IP address: 98.25.83.56.

CHECKIP.COM

Visualizza mappa più grande

Uniti

KANSAS

Lawrence

Kansas City

MI5

Your IP lookup 98.25.83.56
(Change IP-address)

ISP:	Comcast Cable
Organization:	Comcast Cable
Country:	United States

Whats My IP Address?
Your IP is: 98.25.83.56



Use web proxy flexibility to enter the website within my IP hidden, using proxy ip instead

◆ Now we will visit the same site using a proxy web site. For our tests we will use <http://hide.me/en/proxy>

www.whatsmyip.org

Proxy location: Netherlands

- Netherlands
- Germany
- USA

More options

anonymously

URL: <http://www.checkip.com/> [home] [clear cookies]

Options: Encrypt URL Encrypt Page Allow Cookies Remove Scripts Remove Objects

CHECKIP.COM

Your IP lookup 85.17.24.66
(Change IP-address)



Web browser proxy

- ◆ The first piece of information to locate is a proxy address and port.
- ◆ There are many different sites across the internet where proxies are available.
- ◆ In our test we will use one of the proxies listed
 - <https://hidemy.name/en/proxy-list/>

Configure Proxies to Access the Internet

No proxy

Auto-detect proxy settings for this network

Use system proxy settings

Manual proxy configuration:

HTTP Proxy: Port:

Use this proxy server for all protocols

SSL Proxy: Port:

FTP Proxy: Port:

SOCKS Host: Port:

SOCKS v4 SOCKS v5 Remote DNS

No Proxy for:

Example: .mozilla.org, .net.nz, 192.168.1.0/24

Automatic proxy configuration URL:

Do not prompt for authentication if password is saved





Types of HTTP Proxy

◆ There are several sub-types that you can encounter according to the level of anonymity you are looking for:

- High anonymous (Elite)
- Anonymous
- Transparent

destination doesn't know if you're opening from a proxy or client

knows you are using proxy and can see your IP

destination doesn't know my IP or location, only sees the ip of proxy



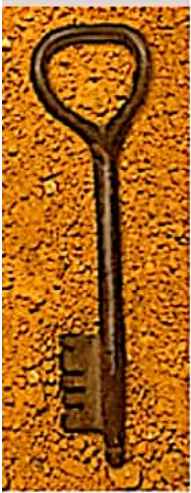
High anonymous (Elite) proxies

- ◆ These proxies do not change request fields and look like they come from a real IP.
- ◆ The users real IP is hidden and there is no indication to the web server that the request is coming from a proxy.



Anonymous proxies

- ◆ These proxy servers also do not show your real IP address, but they do change the request fields.
- ◆ As a result, by analyzing the web log, it is possible to detect that a proxy server was used.
- ◆ Not that this matters however, some server administrators do restrict proxy requests, so they use this type of information to block requests, such as this, in the future.



Transparent proxies

- ◆ Also known as HTTP relay proxies, these systems change the request fields thus, they transfer the real IP address of the user.
- ◆ In other words, these proxy systems offer no security and should therefore never be used for security browsing.
- ◆ The only reason to use these systems, is for network speed improvements.



Proxy choice

- ◆ If you are using HTTP Proxies, be sure to select reliable proxies, preferably ones you own, or you may be exposing your data to an unknown entity, for which you can be held liable.
- ◆ There are also other tools available on the internet that help you in ensuring you know your anonymity is protected.
- ◆ There are many ways to tell that a proxy is a real anonymous proxy.



How to check for real anonymous proxies?

- ◆ A popular way to tell is also to use anonymity testing sites such as:
 - <https://centralops.net/co/>
 - <http://www.nmonitoring.com/>
 - <https://pentest-tools.com/home> my information still available to the proxy
 - <http://do-know.com/privacy-test.html>
 - <http://www.all-nettools.com/>

– Holmes/Who

Are you really anonymous? Check it now!

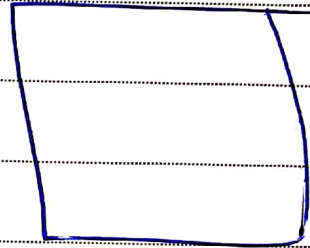
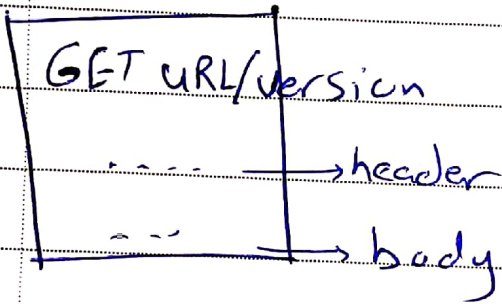
Note: It's completely safe - this script will NOT try to connect to your computer. But you can select additional "ACTIONS" in the generated report to perform port/trojan scanning or get additional information about your network.

[Please read about possible alerts](#)

* HTTP :-

Request

Response



* header contains:

browser, encoding, language &

HTTP_VIA, HTTP_X-Forward_For

- Transparent ← *جسٹریبل* / *جسٹریبل ہے*
- High Anonymus ← not determined *نہیں*
- Anonymus ← IP proxy *IP پراکسی ہے*



HTTP_VIA and HTTP_X_FORWARDED_FOR

- ◆ A standard HTTP communication string would look similar to the following:



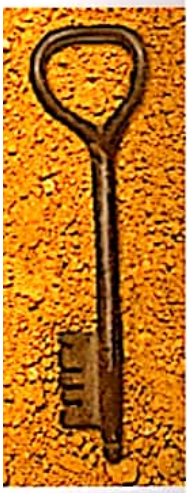
```
REMOTE_ADDR = 98.10.50.155
HTTP_ACCEPT_LANGUAGE = en
HTTP_USER_AGENT = Mozilla/4.0 (compatible; MSIE 5.0; Windows 98)
HTTP_HOST = www.elearnsecurity.com
HTTP_VIA = not determined
HTTP_X_FORWARDED_FOR = not determined
```

- ◆ HTTP_VIA and the HTTP_X_FORWARDED_FOR fields are used by proxy systems in order to show the server that it is acting on behalf of another system.



HTTP_VIA and HTTP_X_FORWARDED_FOR

- ◆ If HTTP_VIA contains an address (or in case of chained proxies, many addresses), it actually indicates that there is a proxy server being used.
- ◆ The IP address included in this field is actually the IP address of the proxy server(s).
- ◆ In contrast, the HTTP_X_FORWARDED_FOR field, if present, indicates the actual IP address of the client that the proxy is acting on behalf of for the communications.



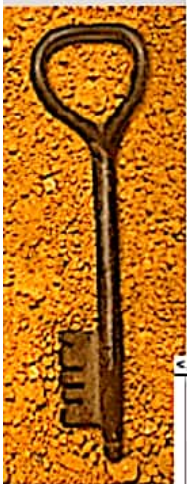
HTTP_VIA and HTTP_X_FORWARD_FOR

- ◆ A simple pass-through or cache proxy communication string would appear as follows:

```
</>
REMOTE_ADDR = 94.89.100.1
HTTP_ACCEPT_LANGUAGE = en
HTTP_USER_AGENT = Mozilla/4.0 (compatible; MSIE 5.0; Windows 98)
HTTP_HOST = www.elearnsecurity.com
HTTP_VIA = 94.89.100.1 (Squid/2.4.STABLE7)
HTTP_X_FORWARDED_FOR = 98.10.50.155
```

Transparent

- ◆ Since we see an IP address in the last two fields, we know that a proxy server is used.
- ◆ By analyzing either web site logs or traffic sniffing files, an administrator can easily find the proxy addresses and in turn, can use these same functions to block further access to the site.



HTTP_VIA and HTTP_X_FORWARD_FOR

- ◆ In the case of high anonymity proxy systems the communication would be similar to the following:

```
</>
REMOTE_ADDR = 94.89.100.1
HTTP_ACCEPT_LANGUAGE = en
HTTP_USER_AGENT = Mozilla/4.0 (compatible; MSIE 5.0; Windows 98)
HTTP_HOST = www.elearnsecurity.com
HTTP_VIA = not determined
HTTP_X_FORWARDED_FOR = not determined
```

High Anonymous

- ◆ The REMOTE_ADDR is actually the address of the proxy system.
- ◆ If this traffic is analyzed, the administrators would have no indication that a proxy system is being used.



Tor network



- ◆ Now that we have covered proxy systems, let us move on to more highly anonymous means of network communication called The Onion Router or Tor.
- ◆ If you want to know more about how tor works, please refer to the following
 - <https://2019.www.torproject.org/about/overview.html.en>



Tor network

- ◆ Tor is a program you can run on your computer that helps keep you safe on the Internet. **like firefox, chrome**
- ◆ It protects you by bouncing your communications around a distributed network of relays run by volunteers all around the world. **→ encryption header between me and intermediate devices**
- ◆ It prevents somebody watching your Internet connection from learning what sites you visit, and it prevents the sites you visit from learning your physical location.
- ◆ This set of volunteer relays is called the **Tor network**.
- ◆ <https://www.torproject.org/docs/faq.html.en>



Why use Tor

- ◆ Traffic analysis can be used to infer who is talking to whom over a public network. It is based on IP header.
- ◆ Even if you encrypt the data payload of your communications, traffic analysis still reveals a great deal about what you're doing and, possibly, what you're saying.
- ◆ That's because it focuses on the header, which discloses source, destination, size, timing, and so on.
- ◆ Authorized intermediaries like Internet service providers, and sometimes unauthorized intermediaries as well can see your traffic.



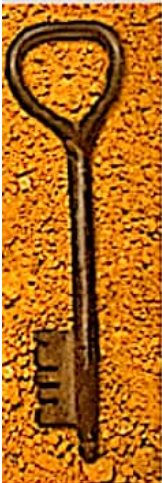
Tor network uses

- ◆ Allows both organizations and individuals to share information over public networks without compromising their privacy.
- ◆ Allows its users to reach otherwise blocked destinations or content.
- ◆ Individuals use Tor to keep websites from tracking them and their family members,
- ◆ Tor's onion services let users publish web sites and other services without needing to reveal the location of the site.
- ◆ Individuals also use Tor for socially sensitive communication: chat rooms and web forums for rape and abuse survivors, or people with illnesses.



Tor network uses ... cont

- ◆ Journalists use Tor to communicate more safely with whistle blowers and dissidents.
- ◆ Non-governmental organizations (NGOs) use Tor to allow their workers to connect to their home website while they're in a foreign country, without notifying everybody nearby that they're working with that organization.



How Tor works?

- ◆ The following is an example of a client operating with Tor:
 - Client requests a list of Tor nodes from a directory server.
 - The client randomly selects nodes on the Tor network (called relays), and encrypts the traffic between each relay. **at least 3**
 - If the client requests a second destination after the specified time limit, another separate tunnel is created for that communication repeating the process.



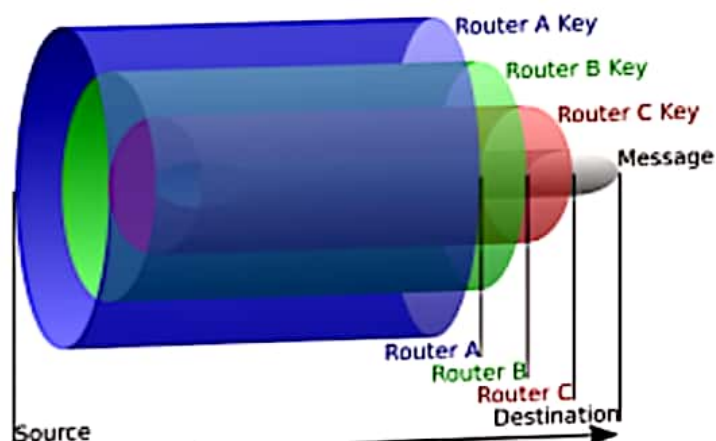
How Tor works ... cont

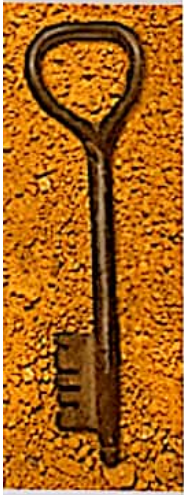
- ◆ The client sends the packet through a series of relays or proxies.
- ◆ The packet is encrypted in multiple layers and routed via multiple hops through the Tor network to the final receiver.
- ◆ Note that all your local ISP can observe now is that you are communicating with Tor nodes.
- ◆ Similarly, servers in the Internet just see that they are being contacted by Tor nodes.
- ◆ Tor Browser can certainly help people access your website in places where it is blocked.



Onion Routing

- ◆ The source sends the encrypted onion to Router A,
- ◆ which removes a layer of encryption to learn only where to send it next and where it came from (though it does not know if the sender is the origin or just another node).
- ◆ Router A sends it to Router B, which decrypts another layer to learn its next destination.
- ◆ Router B sends it to Router C, which removes the final layer of encryption and transmits the original message to its destination.





How is Tor different from other proxies

- ◆ Proxy is paid, no need to install any software.
- ◆ Proxy knows both who you are and what you browse on the Internet.
- ◆ Proxy can see your traffic as it passes through their server.
- ◆ You have to trust the provider isn't watching your traffic, injecting their own advertisements into your traffic stream, or recording your personal details.
- ◆ Tor passes your traffic through at least 3 different servers before sending it on to the destination.
- ◆ Because there's a separate layer of encryption for each of the three relays, somebody watching your Internet connection can't modify, or read, what you are sending into the Tor network.
- ◆ Your traffic is encrypted between the Tor client (on your computer) and where it pops out somewhere else in the world.



Tor Security

- ◆ Tor uses a variety of different keys, with three goals in mind:
 1. encryption to ensure privacy of data within the Tor network,
 2. authentication so clients know they're talking to the relays they meant to talk to, and
 3. signatures to make sure all clients know the same set of relays.



Tor Encryption

- ◆ All connections in Tor use TLS link encryption, **SSL**
 - ◆ So observers can't look inside to see which circuit a given cell is intended for.
 - ◆ Further, the Tor client establishes an ephemeral encryption key with each relay in the circuit; these extra layers of encryption mean that only the exit relay can read the cells.
 - ◆ Both sides discard the circuit key when the circuit ends, so logging traffic and then breaking into the relay to discover the key won't work.
- if i encrypted the original cell "message" even the exit relay cannot read it



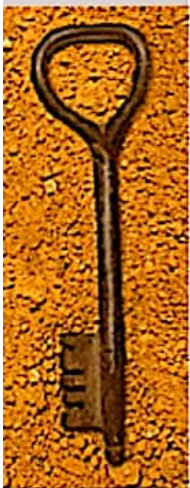
Tor Authentication

- ◆ Every Tor relay has a public decryption key called the "onion key". **certificate signed by TOR directory**
- ◆ Each relay rotates its onion key once a week.
- ◆ When the Tor client establishes circuits, at each step it demands that the Tor relay prove knowledge of its onion key.
- ◆ That way the first node in the path can't just spoof the rest of the path. Because the Tor client chooses the path, it can make sure to get Tor's "distributed trust" property.
- ◆ No single relay in the path can know about both the client and what the client is doing.



Tor Coordination

- ◆ How do clients know what the relays are, and How do they know that they have the right keys for them?
 - Each relay has a long-term public signing key called the "identity key".
 - Each directory authority additionally has a "directory signing key".
 - The directory authorities provide a signed list of all the known relays, and in that list are a set of certificates from each relay specifying their keys, locations, exit policies, and so on.

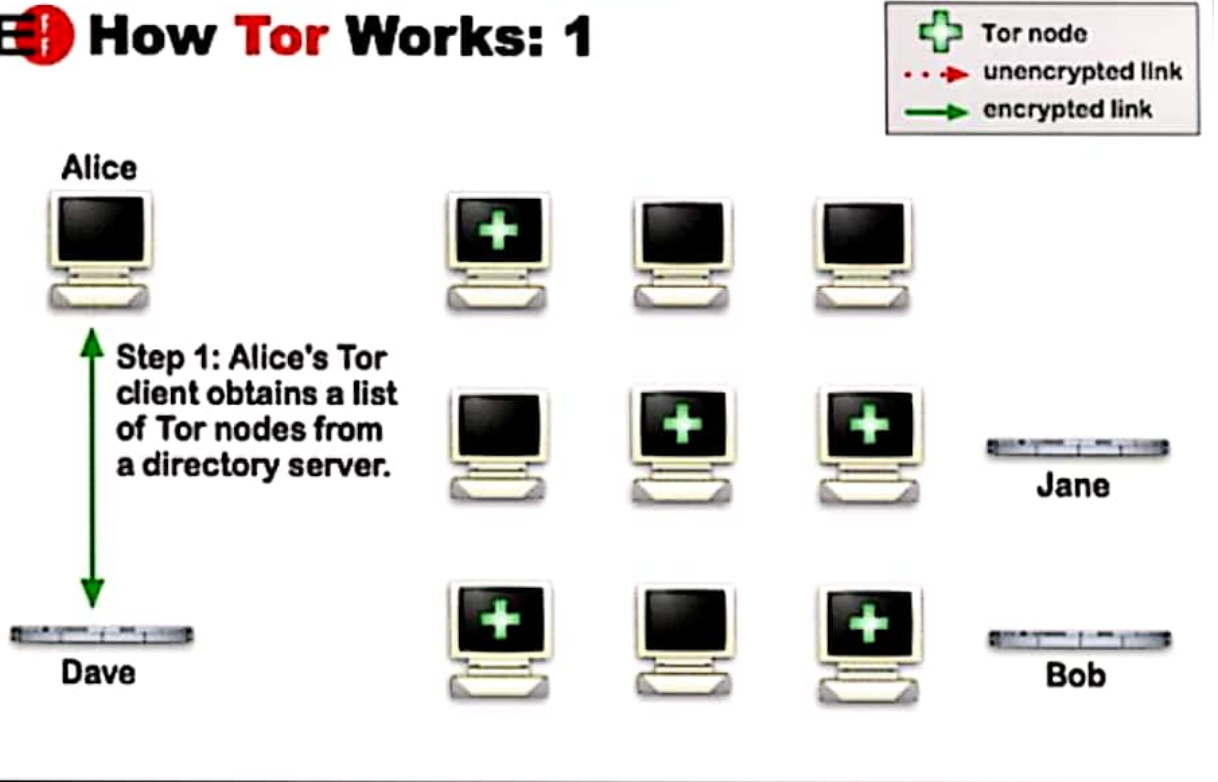


How client knows directory authorities

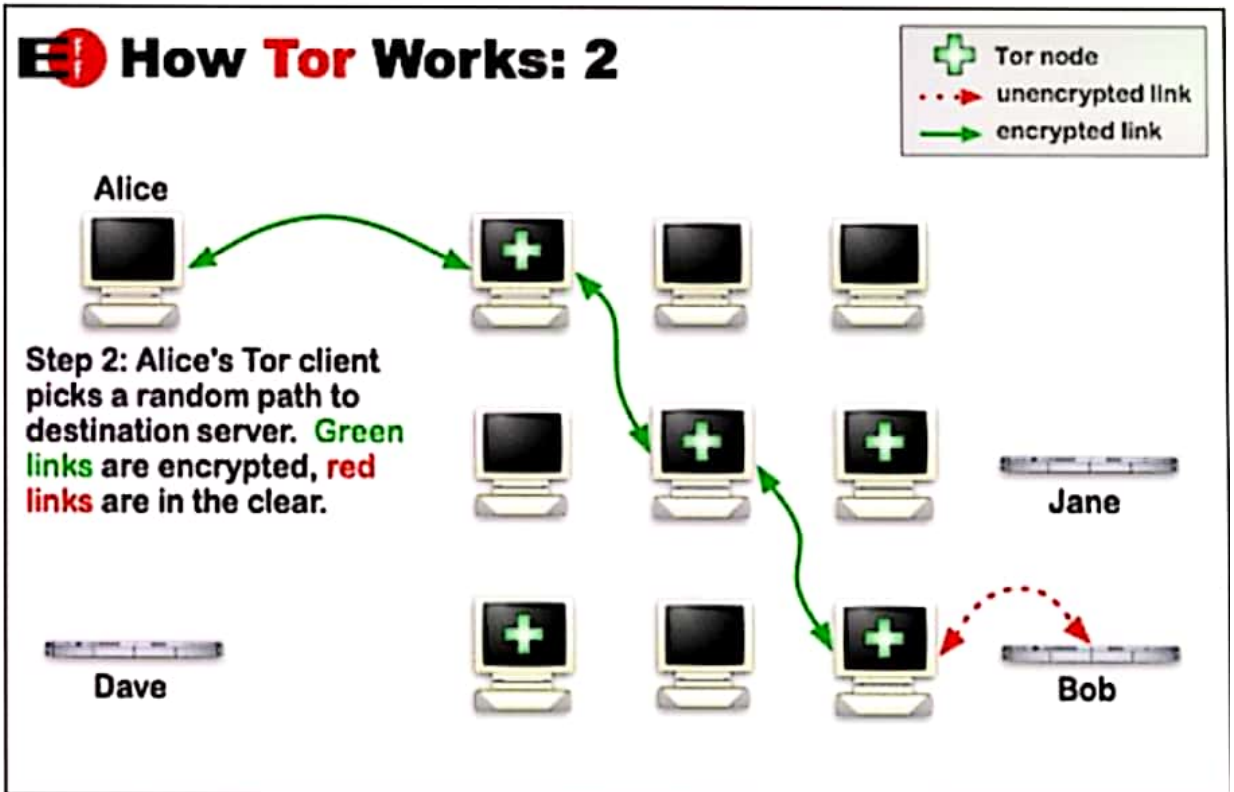
- ◆ The Tor software comes with a built-in list of location and public key for each directory authority.
- ◆ So the only way to trick users into using a fake Tor network is to give them a specially modified version of the software.
- ◆ How do users know they've got the right software?
- ◆ When Tor source code or package is distributed, it is signed with GNU Privacy Guard.



How Tor Works: 1

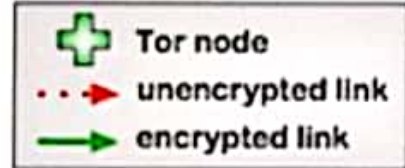


How Tor Works: 2





How Tor Works: 3



Alice



Step 3: If at a later time, the user visits another site, Alice's tor client selects a second random path. Again, green links are encrypted, red links are in the clear.



Dave



Jane



Bob