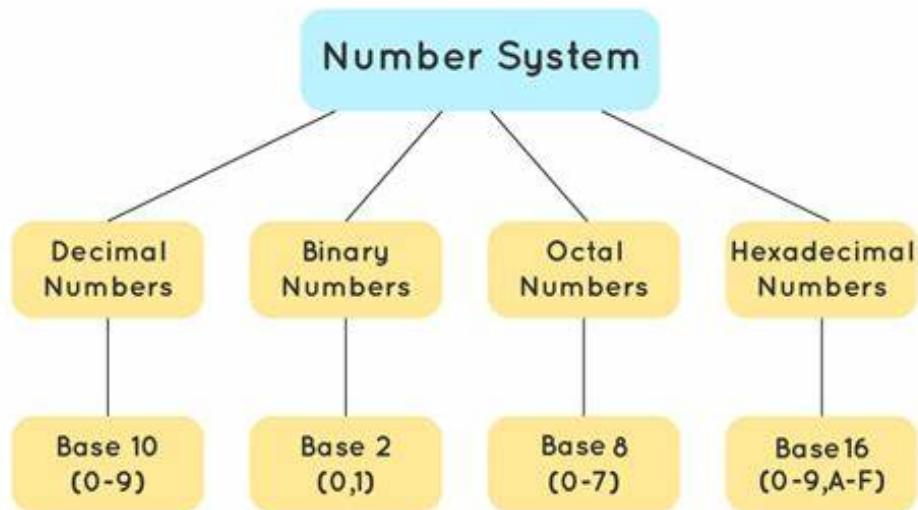


Computer Concept

Number System

Types of Number System

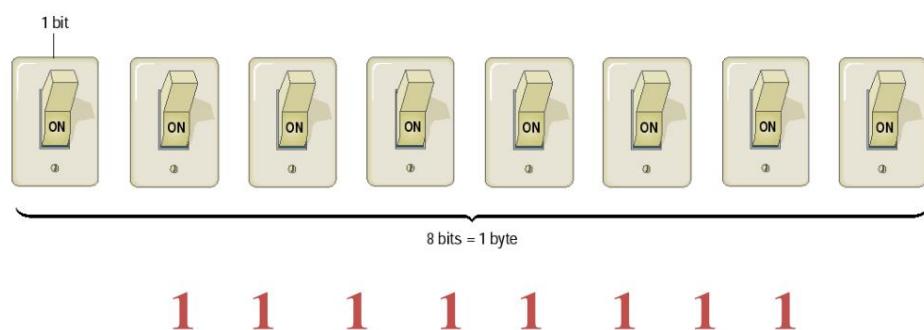


Binary Number System

Number System

Bits and Bytes

- A single unit of data is called a bit, having a value of 1 or 0.
- Computers work with collections of bits, grouping them to represent larger pieces of data, such as letters of the alphabet.
- Eight bits make up one byte. A byte is the amount of memory needed to store one alphanumeric character.
- With one byte, the computer can represent one of 256 different symbols or characters.



Base Conversion

Number Systems Conversion Chart

Binary																
Place	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}
Weight	2048	1024	512	256	128	64	32	16	8	4	2	1	0.5	0.25	0.125	0.0625

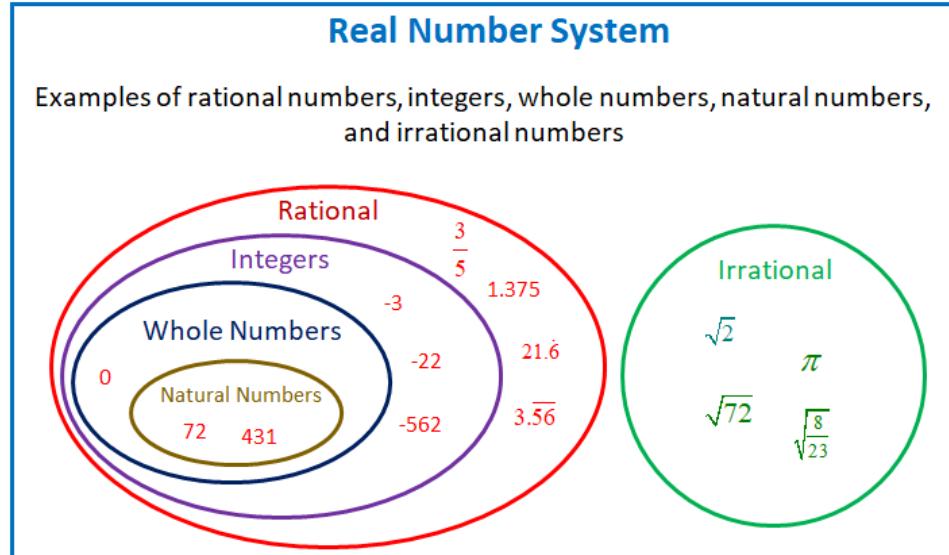
Binary, Hex, and Octal Conversions

Binary	Octal	Hexadecimal	Decimal
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	10	8	8
1001	11	9	9
1010	12	A	10
1011	13	B	11
1100	14	C	12
1101	15	D	13
1110	16	E	14
1111	17	F	15

Methodology

- Convert from decimal to binary
Divide the decimal by the largest binary weight it is divisible by and place a "1" in that column. Then select the next largest weight, if it is divisible put a "1" in that column otherwise place a "0" in the column. Continue until all the columns have either a "1" or "0" resulting in a binary expression.
- Convert from decimal to hex.
Convert to binary first, then group the binary in groups of 4 beginning on the right working to the left. For each group determine the hex value based on the table to the left.
- Convert to octal
Convert to binary first, then group the binary in groups of 3 beginning on the right working to the left. For each group determine the octal value based on the table to the left.

Real Number System



Complements

Complements of numbers

(r-1)'s Complement

- Given a number N in base r having n digits,
- the $(r-1)$'s complement of N is defined as

$$(r^n - 1) - N$$

- For decimal numbers the base or $r = 10$ and $r-1 = 9$,

- so the 9's complement of N is $(10^n - 1) - N$

- $99999\dots\dots - N$

9	9	9	9	9
Digit n	Digit n-1	Next digit	Next digit	First digit

Complements

- There are two types of complements for each base- r system: the radix complement and diminished radix complement.

→ the r 's complement and the second as the $(r-1)$'s complement.

Diminished Radix Complement

Given a number N in case r having n digits, the $(r-1)$'s complement of N is defined as $(r^n - 1) - N$. For decimal numbers, $r = 10$ and $r-1 = 9$, so the 9's complement of N is $(10^n - 1) - N$.

Example:

The 9's complement of 546700 is $999999 - 546700 = 453299$.

The 9's complement of 012398 is $999999 - 012398 = 987601$.

- For binary numbers, $r = 2$ and $r-1 = 1$, so the 1's complement of N is $(2^n - 1) - N$.

Example:

The 1's complement of 1011000 is 0100111

The 1's complement of 0101101 is 1010010

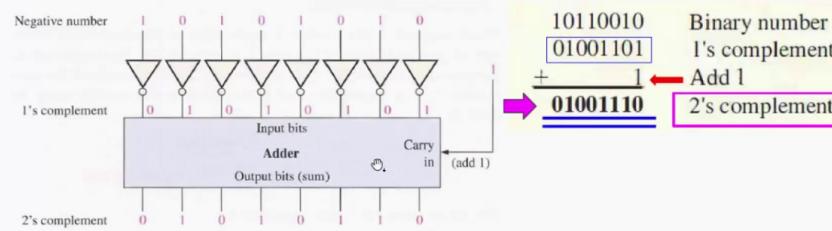
2.5) 1 'S and 2'S Complements of Binary Numbers

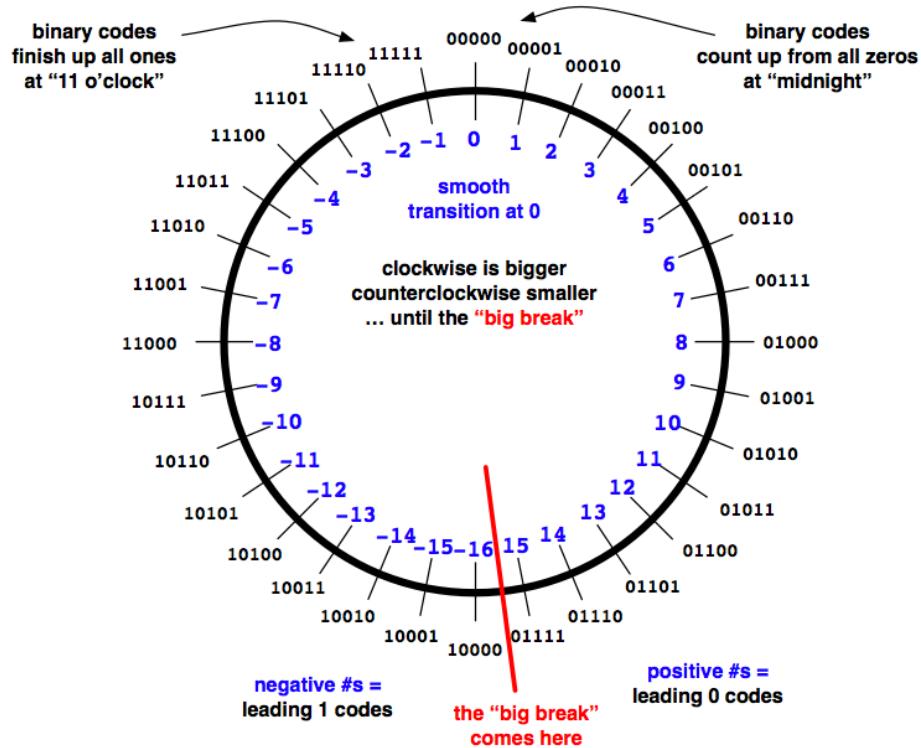
Finding the 2 's Complement

The 2's complement of a binary number is found by adding it to the LSB of the 1's complement.

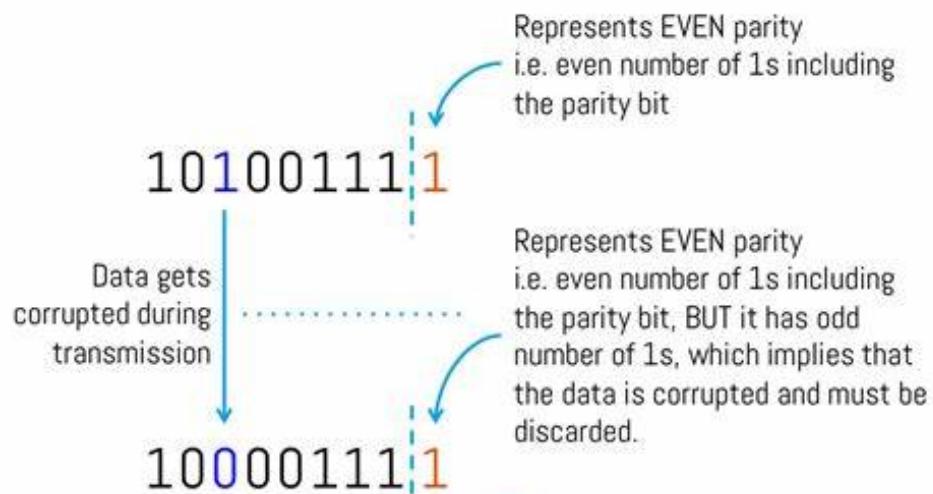
$$2\text{'s complement} = (\text{1's complement}) + 1$$

Example 17: Find the 2's Complement of 10110010

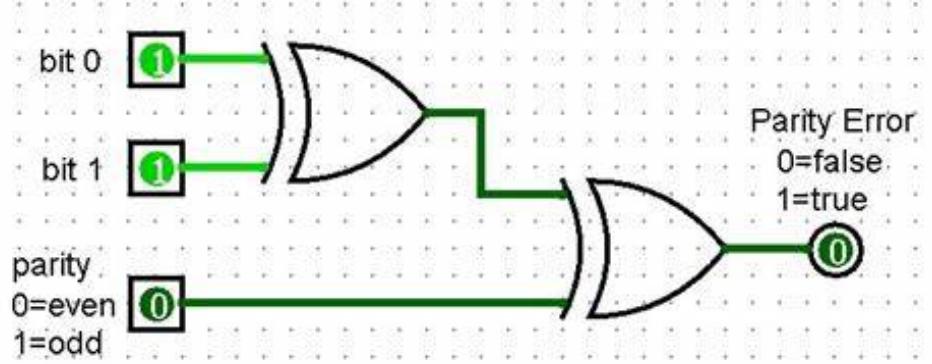




Parity Bit

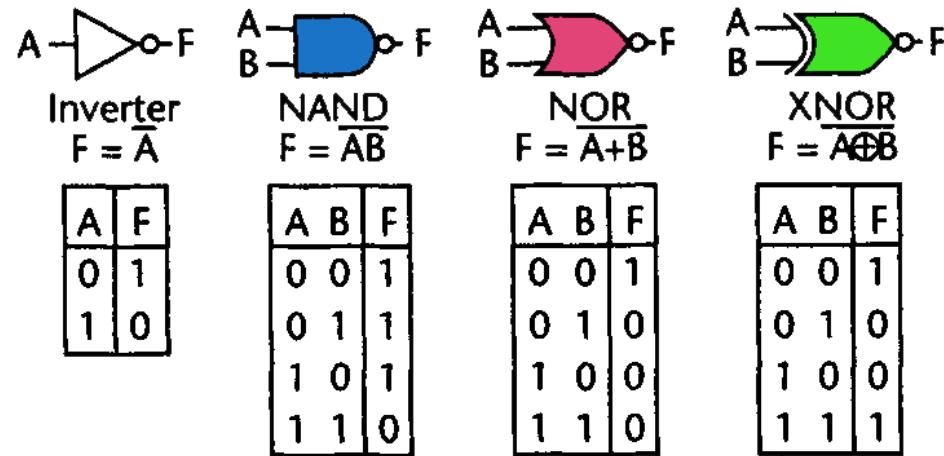
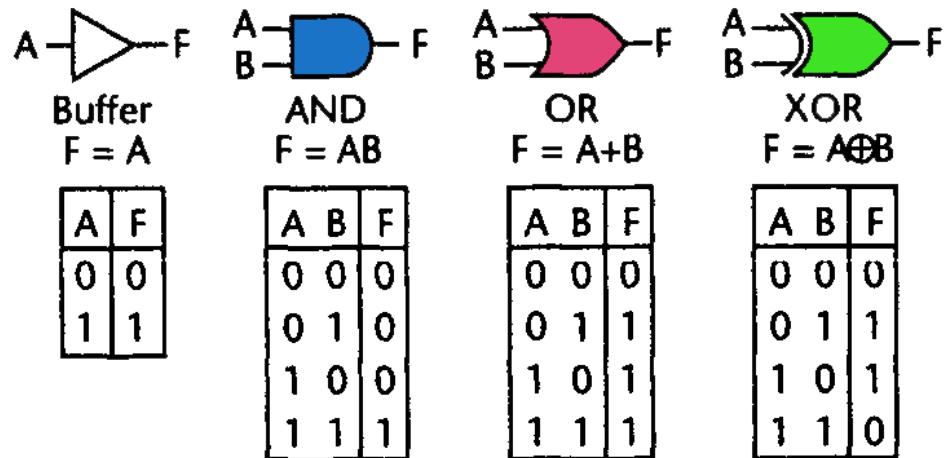


Why do we need Parity Bit?
© maxEmbedded.com 2013

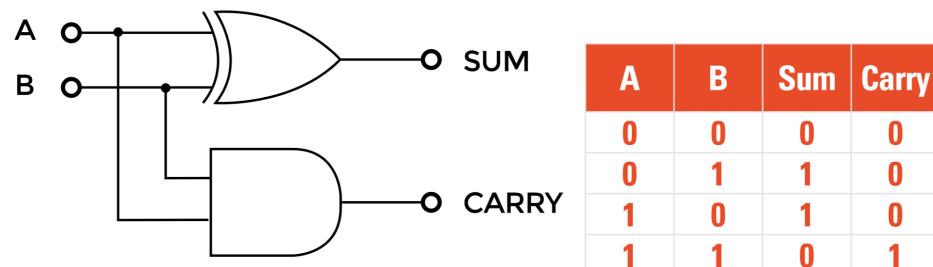


Logic Design

Logic Gates

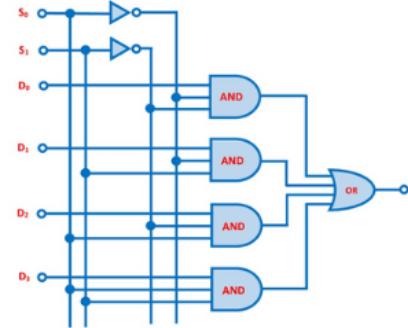
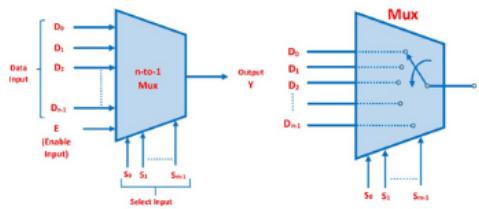


Half-Adder



Multiplexer

What is a Multiplexer?



Electrical 4 U

Diodes

Symbol	Name	Physical Look	Applications
	Diode		Mostly used as rectifier for converting ac to dc
	Zener Diode		Used as a constant voltage source
	Schottky Diode		High current diode used for high current
	Tunnel Diode		Used in oscillator and microwave amplifier
	Light Emitting Diode (LED)		Emit light when it is forward biased
	Photodiode		Allows reverse current when light falls on it
	Laser Diode		Used in CD players
	Current Regulator Diode		Maintains constant current
	Varactor Diode		Acts as variable capacitor in reverse bias
	Pin diode		Acts as capacitor in reverse and variable resistor in forward

Data Presentation and Processing

Machine Language and Assembly Language

Machine Language

- Instructions, like registers and words of data, are also 32 bits long
 - Example: add \$t0, \$s1, \$s2
 - registers have numbers, \$t0=8, \$s1=17, \$s2=18
- Instruction Format:

000000	10001	10010	01000	00000	100000
op	rs	rt	rd	shamt	funct

- Can you guess what the field names stand for?**

op = Basic operation of the instruction: opcode

rs = The first register source operand

rt = The second register source operand

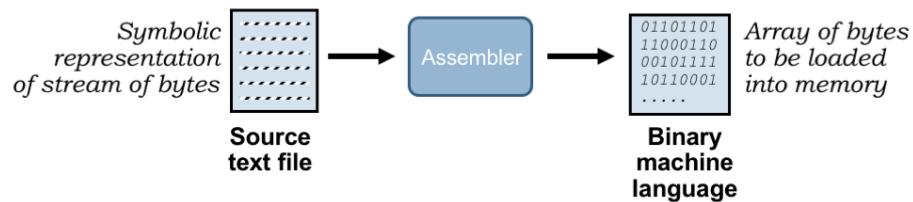
rd = The register destination operand

shamt = shift amount

funct = function code

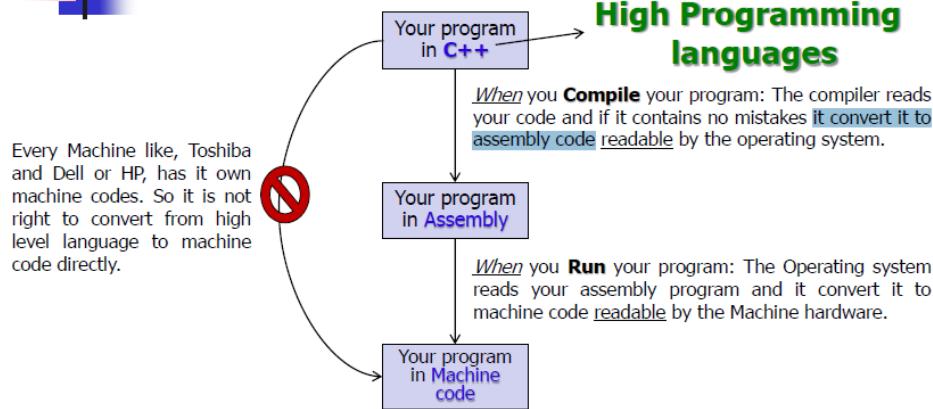
25

Assembly Language



- Abstracts bit-level representation of instructions and addresses
- We'll learn UASM ("microassembler"), built into BSim
- Main elements:
 - Values
 - Symbols
 - Labels (symbols for addresses)
 - Macros

Compiling and running your program



ASCII

ASCII control characters		ASCII printable characters								Extended ASCII characters							
00	NULL (Null character)	32	space	64	@	96	`	128	ç	160	á	192	l	224	ó		
01	SOH (Start of Header)	33	!	65	A	97	a	129	ü	161	í	193	ł	225	ó		
02	STX (Start of Text)	34	"	66	B	98	b	130	é	162	ó	194	ł	226	ó		
03	ETX (End of Text)	35	#	67	C	99	c	131	à	163	ú	195	ł	227	ó		
04	EOT (End of Trans.)	36	\$	68	D	100	d	132	à	164	ñ	196	—	228	ó		
05	ENQ (Enquiry)	37	%	69	E	101	e	133	à	165	N	197	+	229	ó		
06	ACK (Acknowledgement)	38	&	70	F	102	f	134	à	166	”	198	á	230	µ		
07	BEL (Bell)	39	'	71	G	103	g	135	ç	167	º	199	A	231	þ		
08	BS (Backspace)	40	(72	H	104	h	136	è	168	ż	200	ł	232	þ		
09	HT (Horizontal Tab)	41)	73	I	105	i	137	ë	169	®	201	ł	233	ú		
10	LF (Line feed)	42	*	74	J	106	j	138	è	170	¬	202	ł	234	ó		
11	VT (Vertical Tab)	43	+	75	K	107	k	139	í	171	½	203	ł	235	ú		
12	FF (Form feed)	44	,	76	L	108	l	140	í	172	¼	204	ł	236	ý		
13	CR (Carriage return)	45	-	77	M	109	m	141	í	173	í	205	=	237	ý		
14	SO (Shift Out)	46	.	78	N	110	n	142	ä	174	«	206	ł	238	—		
15	SI (Shift In)	47	/	79	O	111	o	143	ä	175	»	207	ł	239	—		
16	DLE (Data link escape)	48	0	80	P	112	p	144	É	176	ł	208	ð	240	≡		
17	DC1 (Device control 1)	49	1	81	Q	113	q	145	æ	177	ł	209	đ	241	±		
18	DC2 (Device control 2)	50	2	82	R	114	r	146	Æ	178	ł	210	đ	242	≡		
19	DC3 (Device control 3)	51	3	83	S	115	s	147	ö	179	ł	211	đ	243	÷		
20	DC4 (Device control 4)	52	4	84	T	116	t	148	ö	180	ł	212	đ	244	¶		
21	NAK (Negative acknowl.)	53	5	85	U	117	u	149	ö	181	ä	213	ı	245	§		
22	SYN (Synchronous idle)	54	6	86	V	118	v	150	ø	182	ä	214	ı	246	+		
23	ETB (End of trans. block)	55	7	87	W	119	w	151	ù	183	ä	215	ı	247	·		
24	CAN (Cancel)	56	8	88	X	120	x	152	ÿ	184	ø	216	ı	248	·		
25	EM (End of medium)	57	9	89	Y	121	y	153	ö	185	ł	217	ł	249	·		
26	SUB (Substitute)	58	:	90	Z	122	z	154	ü	186	ł	218	ł	250	·		
27	ESC (Escape)	59	;	91	[123	{	155	ø	187	ł	219	ñ	251	‘		
28	FS (File separator)	60	<	92	\	124		156	£	188	ł	220	ñ	252	‘		
29	GS (Group separator)	61	=	93]	125	}	157	ø	189	¢	221	—	253	‘		
30	RS (Record separator)	62	>	94	^	126	~	158	×	190	¥	222	—	254	■		
31	US (Unit separator)	63	?	95	—			159	f	191	ł	223	—	255	nbsp;		
127	DEL (Delete)							160		192		224		256			

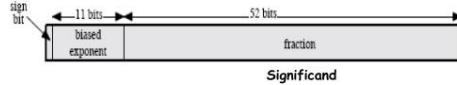
Precision floating-point

Double Precision Floating Point Numbers IEEE Standard

64 bit Double Precision Floating Point Numbers are stored as:

S EEEEEEEEEE FFFFFFFF...FFFF...FFFF...FFFF...FFFF...FFFF...FFFF...FFFF

S: Sign - 1 bit
E: Exponent - 11 bits
F: Fraction - 52 bits



The value V:

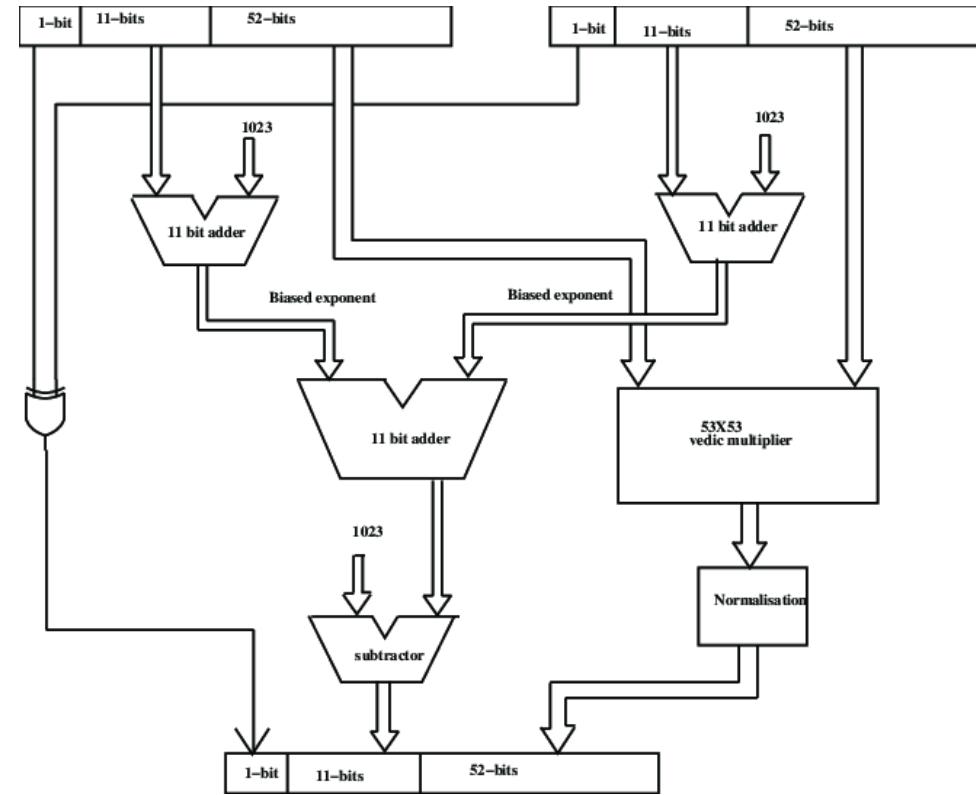
- If E=2047 and F is nonzero, then V= NaN ("Not a Number")
- If E=2047 and F is zero and S is 1, then V= - Infinity
- If E=2047 and F is zero and S is 0, then V= Infinity
- If 0<E<2047 then V= (-1)**S * 2 ** (E-1023) * (1.F) (exponent range = -1023 to +1024)
- If E=0 and F is nonzero, then V= (-1)**S * 2 ** (-1022) * (0.F) ("unnormalized" values)
- If E=0 and F is zero and S is 1, then V= - 0
- If E=0 and F is zero and S is 0, then V= 0

Note: 2047 decimal = 1111111111 in binary (11 bits)

Biased Exponent - Cont'd

- ❖ For double precision, exponent field is 11 bits
 - ◊ E can be in the range 0 to 2047
 - ◊ E = 0 and E = 2047 are reserved for special use
 - ◊ E = 1 to 2046 are used for normalized floating point numbers
 - ◊ Bias = 1023 (half of 2046), val(E) = E - 1023
 - ◊ val(E=1) = -1022, val(E=1023) = 0, val(E=2046) = 1023
- ❖ Value of a Normalized Floating Point Number is

$$\begin{aligned}
 & (-1)^S \times (1.F)_2 \times 2^{E - \text{Bias}} \\
 & (-1)^S \times (1.f_1 f_2 f_3 f_4 \dots)_2 \times 2^{E - \text{Bias}} \\
 & (-1)^S \times (1 + f_1 \times 2^{-1} + f_2 \times 2^{-2} + f_3 \times 2^{-3} + f_4 \times 2^{-4} \dots)_2 \times 2^{E - \text{Bias}}
 \end{aligned}$$



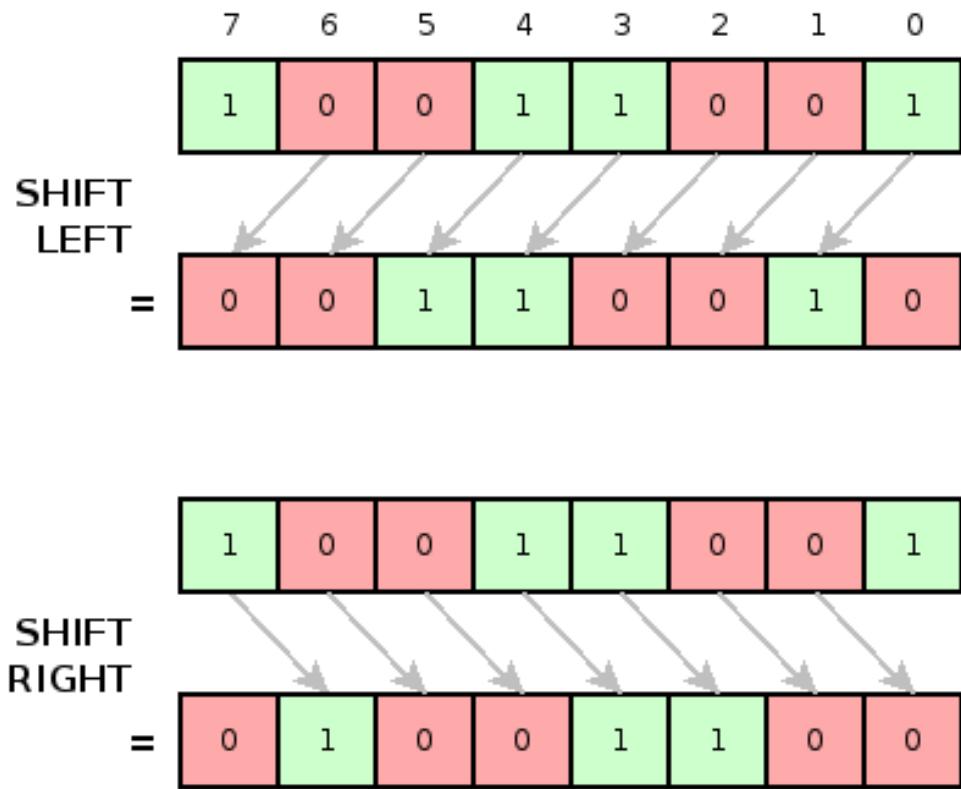
Bitwise Operation

Bitwise Operators

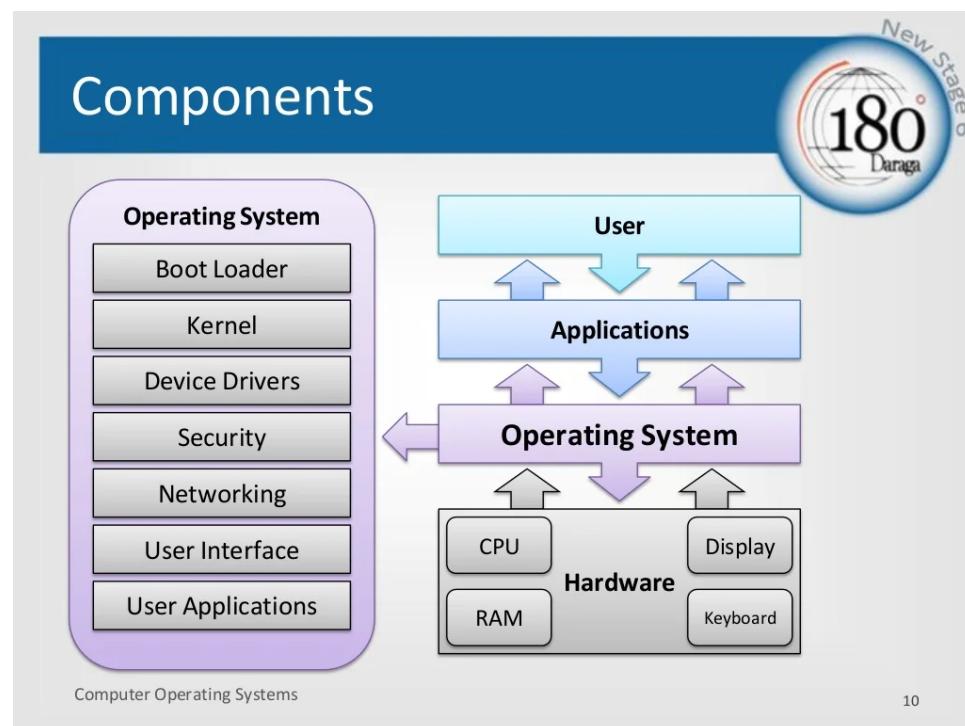
int a = 10, b = 2 for all examples below

Operator	Meaning	Example	Result
<code>~</code>	Bitwise unary NOT	<code>~a</code>	-11
<code>&</code>	Bitwise AND	<code>a&b</code>	2
<code> </code>	Bitwise OR	<code>a b</code>	10
<code>^</code>	Bitwise Ex-OR	<code>a^b</code>	8
<code>>></code>	Shift right	<code>a>>1</code>	5
<code>>>></code>	Shift right zero fill	<code>a>>>1</code>	5
<code><<</code>	Shift left	<code>a<<1</code>	20
<code>&=</code>	Bitwise AND assignment	<code>a &= b</code>	2
<code> =</code>	Bitwise OR assignment	<code>a = b</code>	10
<code>^=</code>	Bitwise Ex-OR assignment	<code>a ^= b</code>	8
<code>>>=</code>	Shift right assignment	<code>a >>= 1</code>	5
<code>>>>=</code>	Shift right zero fill assignment	<code>a >>>= 1</code>	5
<code><<=</code>	Shift left assignment	<code>a <<= 1</code>	20

Startertutorials.com



Operating System



Types of OS

What are the different types?

Mac OS is a series of graphical user interface-based operating systems developed by Apple Inc. for their Macintosh



Linux is a Unix-like computer operating system assembled under the model of free and open source software development and distribution.



Microsoft Windows is a series of graphical interface operating systems developed, marketed, and sold by Microsoft.



iOS (previously iPhone OS) is a mobile operating system developed and distributed by Apple Inc. Originally unveiled in 2007 for the iPhone, it has been extended to support other Apple devices such as the iPod Touch

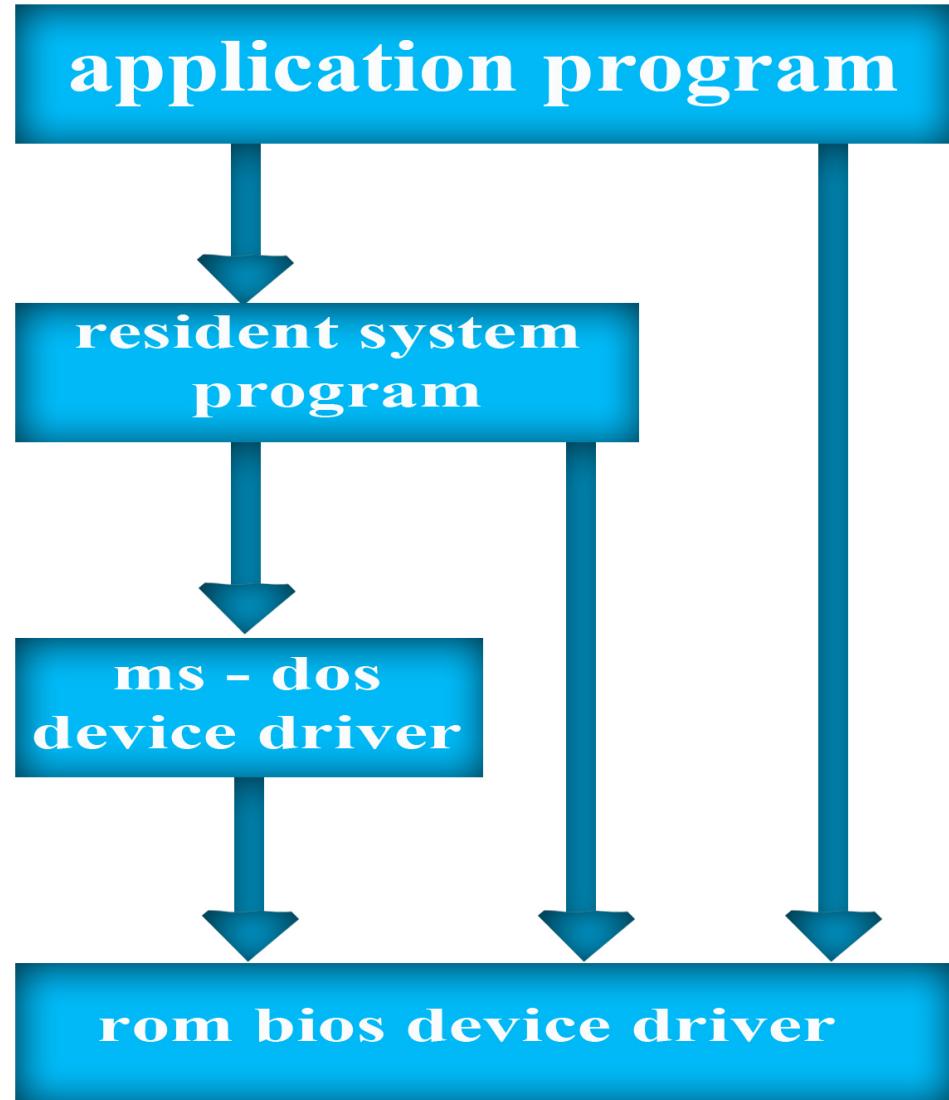


Android is a Linux-based operating system designed primarily for touchscreen mobile devices such as smartphones and tablet computers. Initially developed by Android, Inc.



BSD/OS had a reputation for reliability in server roles; the renowned Unix programmer and author W. Richard Stevens used it for his own personal web server for this reason.

OS Structure



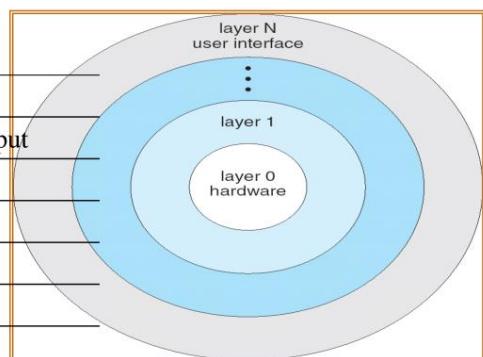
ms dos system structure

Layered Structure of the THE OS

- A layered design was first used in THE operating system.

Its six layers are as follows:

- layer 5: user programs
- layer 4: buffering for input and output
- layer 3: Process management
- layer 2: memory management
- layer 1: CPU scheduling
- layer 0: hardware



Operating System Layers

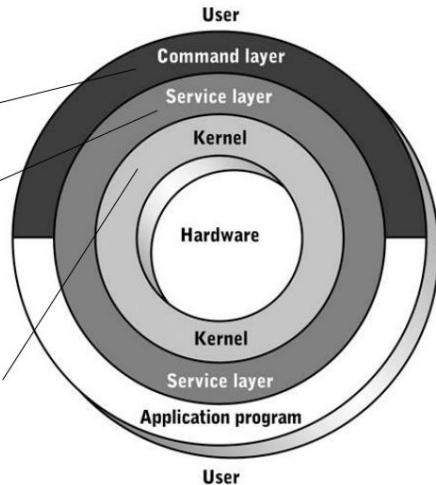
Figure 11-3 ►

Operating system layers (shaded)

User's interface to OS

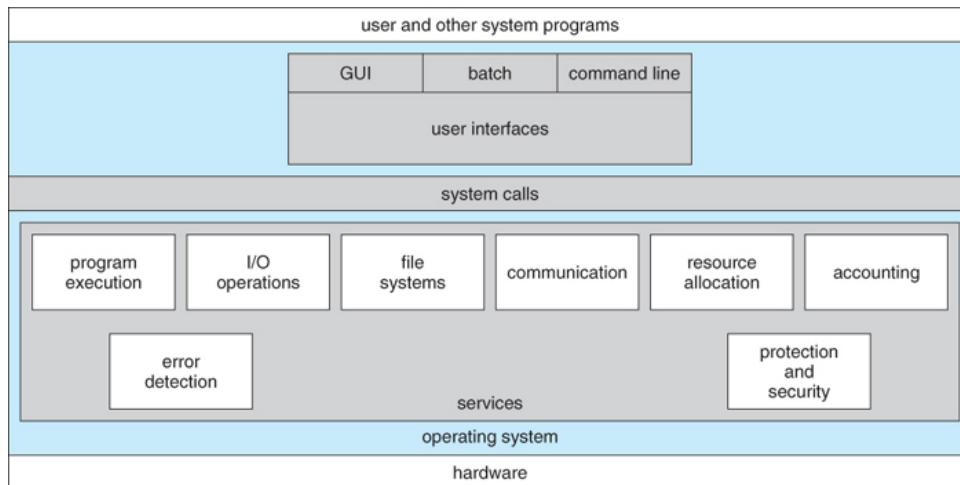
Contains set of functions executed by application programs and command layer

Manages resources; interacts directly with computer hardware

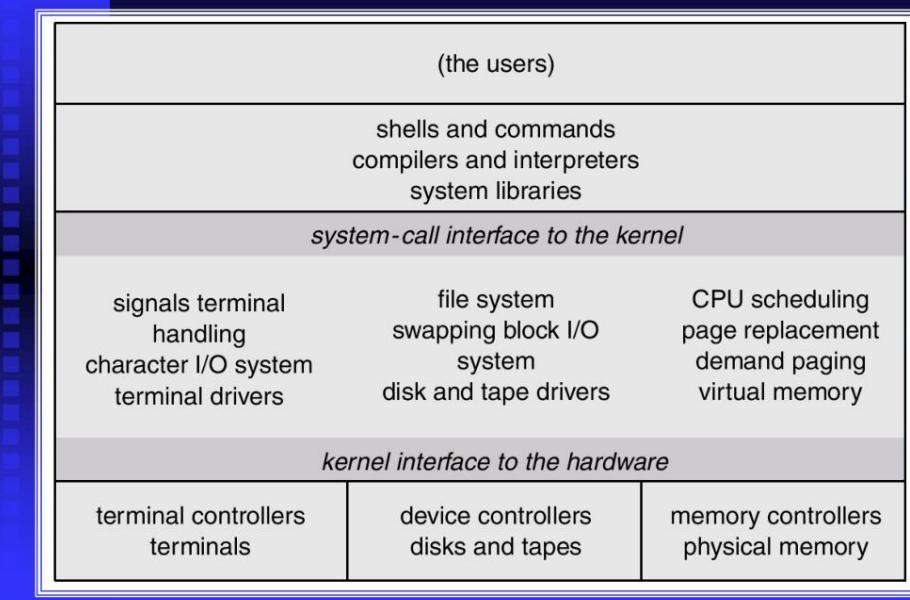


Hardware and Software Architecture

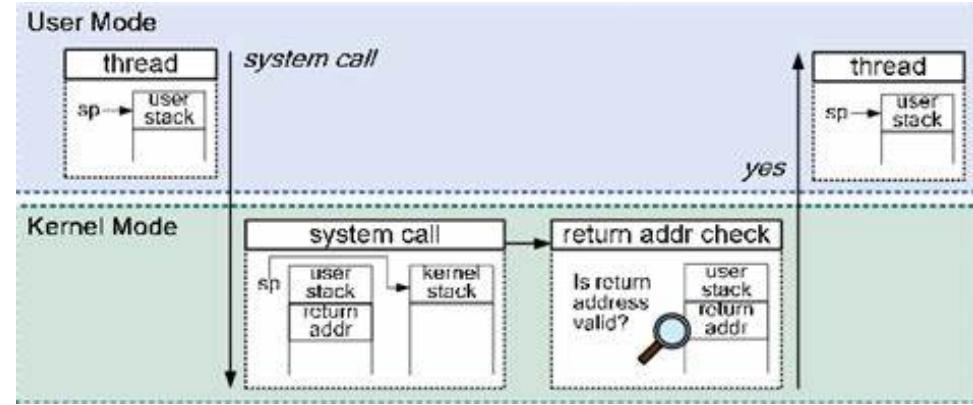
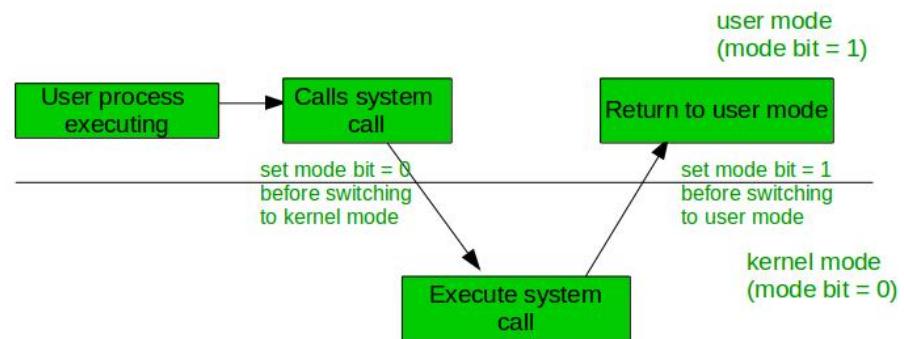
16



OS Structure: Monolithic structure



Dual Mode Operation



User Interface

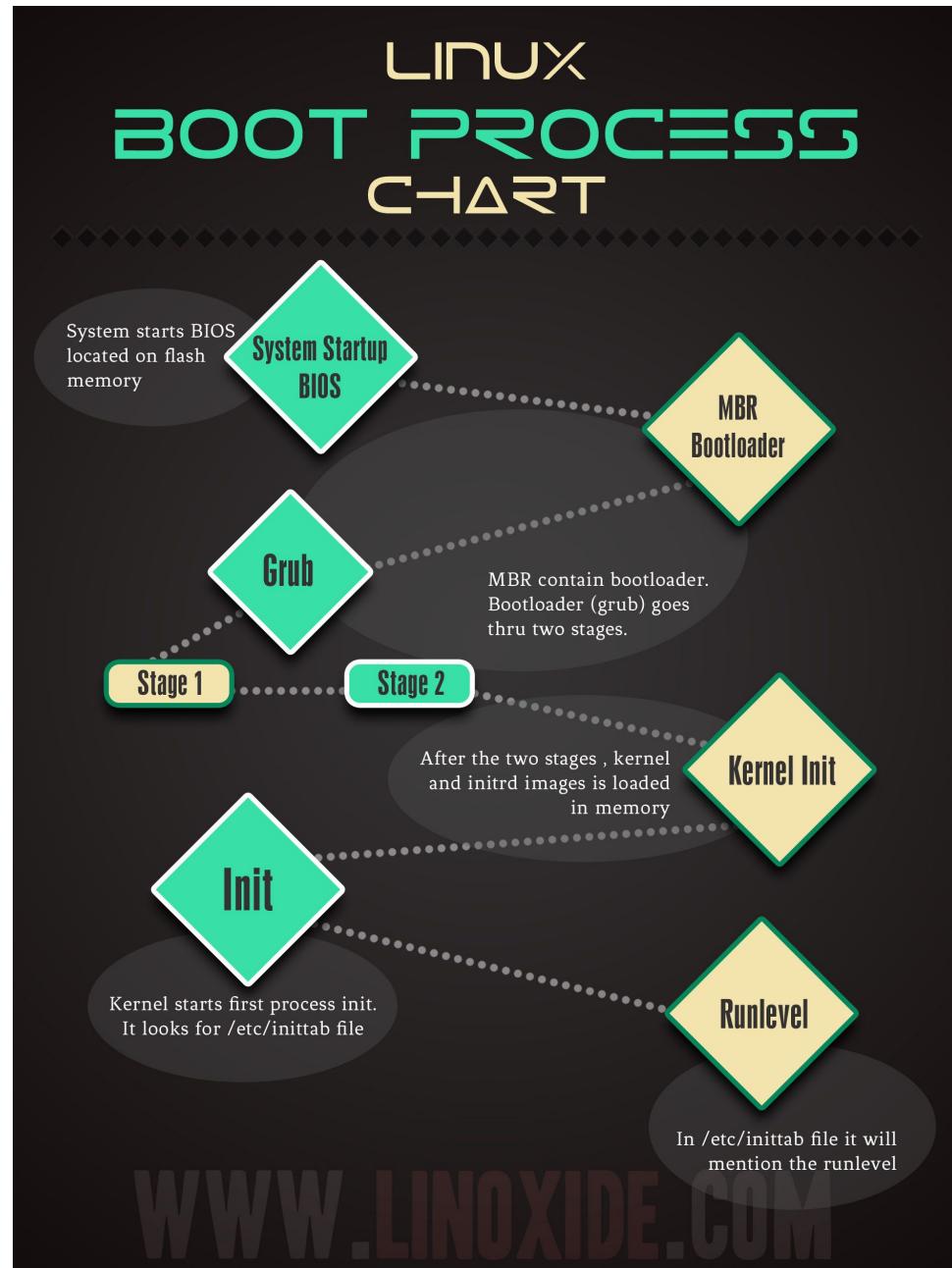
1. User Interface

- Command Line Interface (CLI)
 - Interaction with text/commands
- Batch Files
 - Interaction with the help of batch files (.bat)
- Graphical User Interface (GUI)
 - User friendly
 - Easier to use : pointing device, menus etc
- Hybrid UI

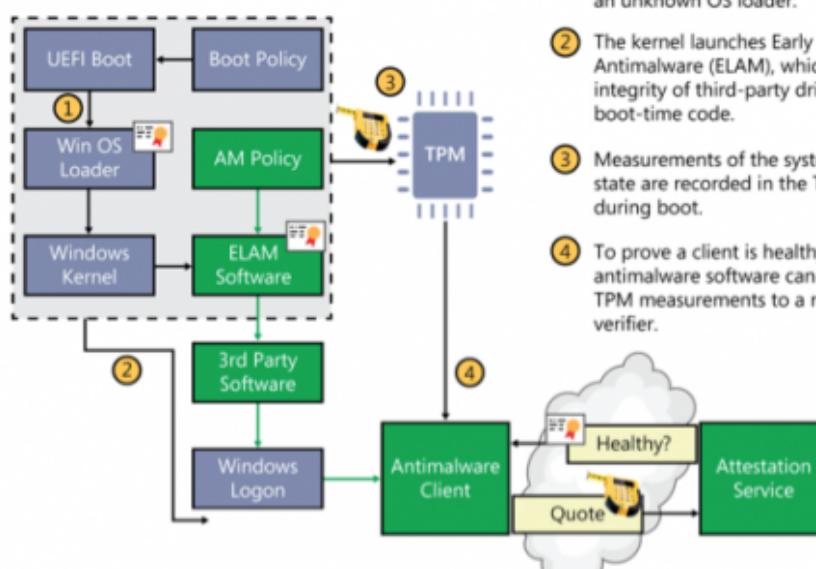


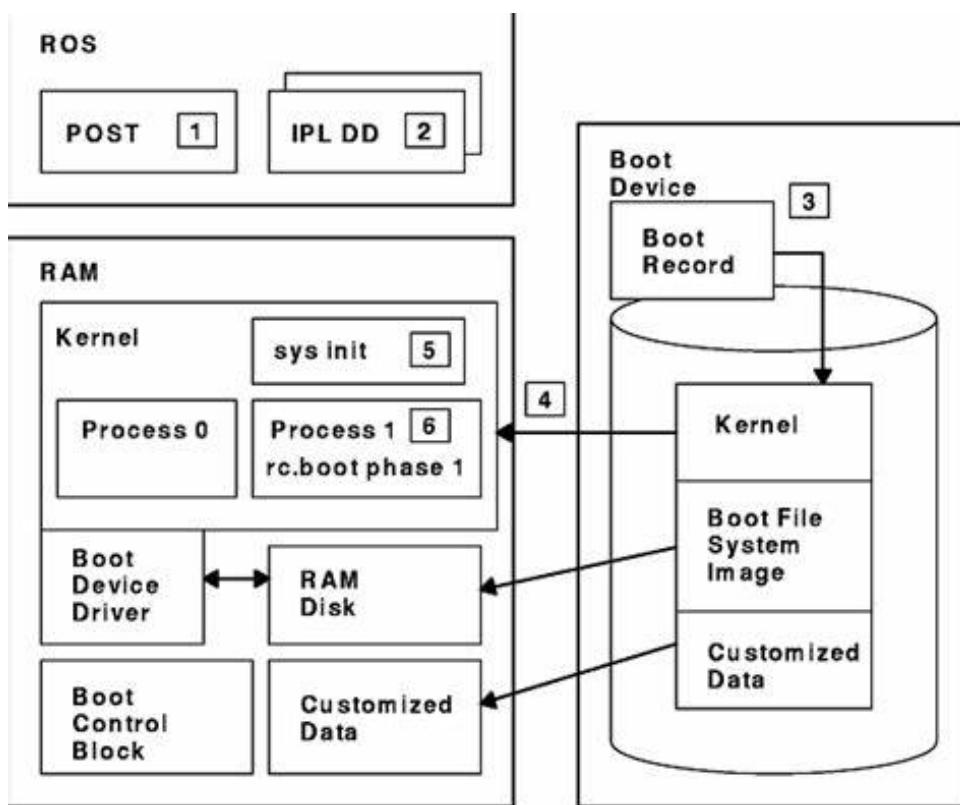
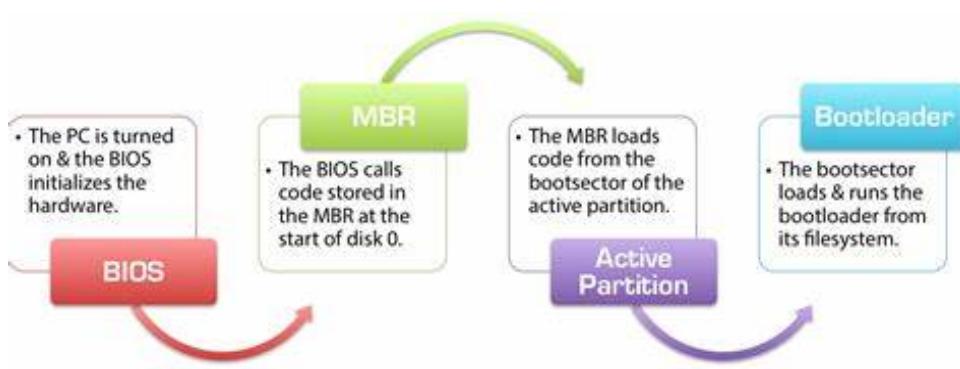
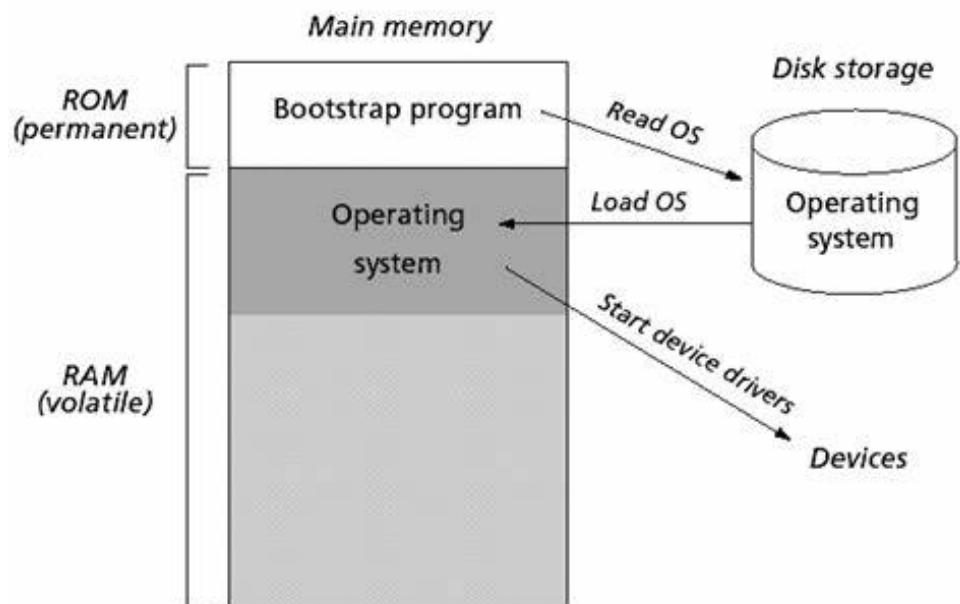
S

System Boot

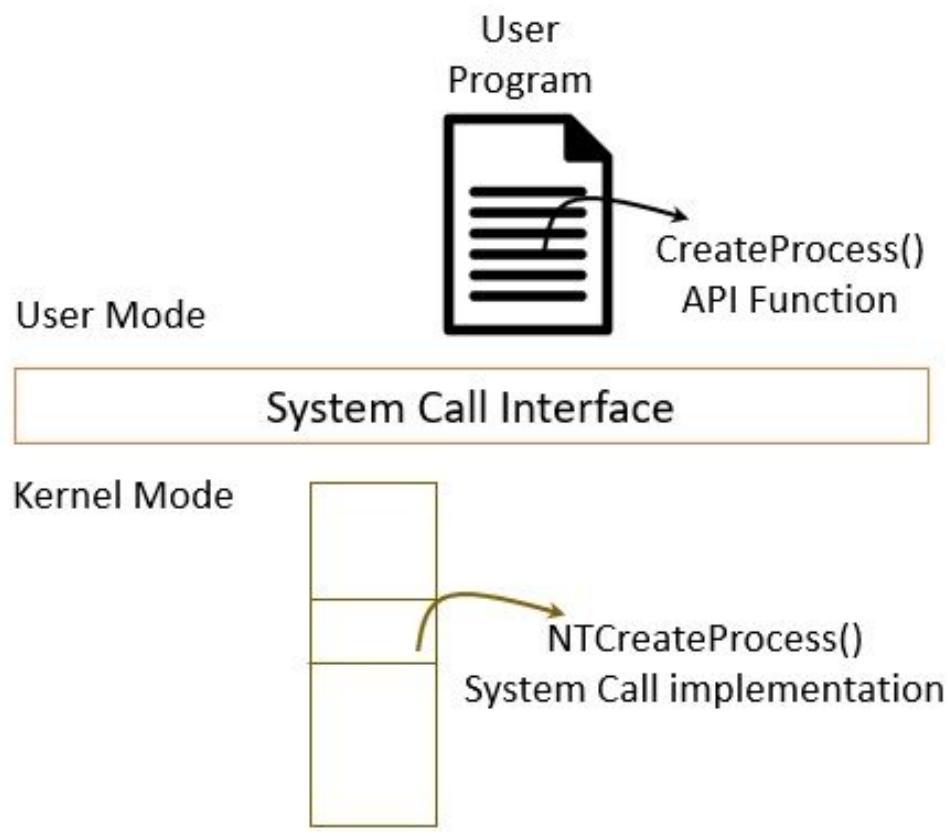
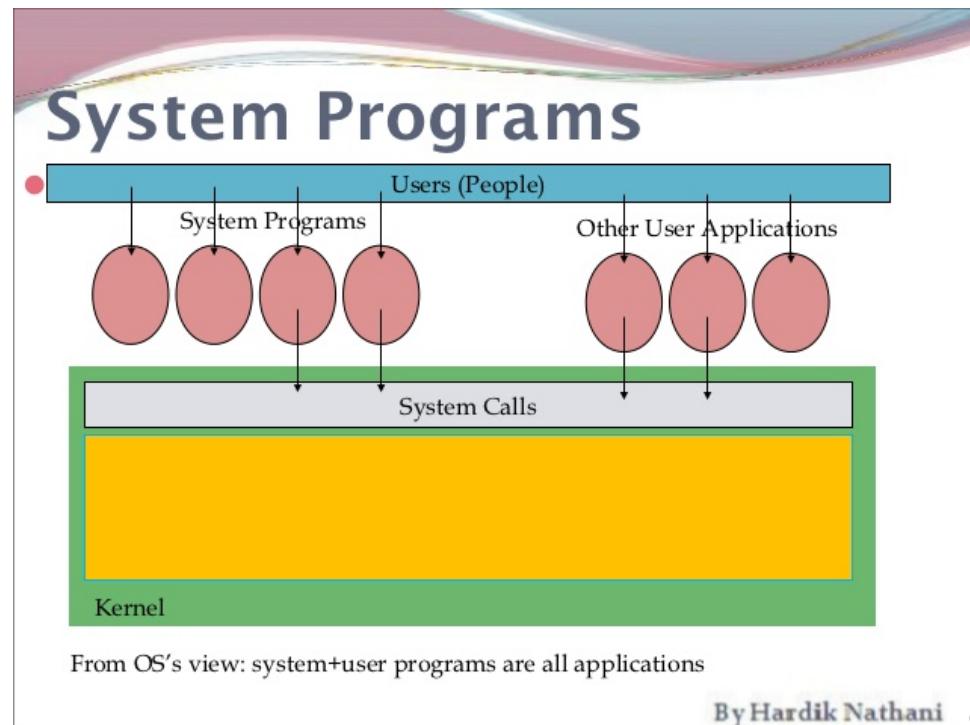


(Windows 8.1 and later)

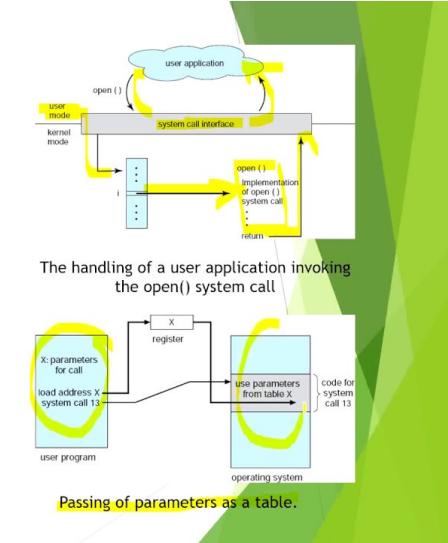
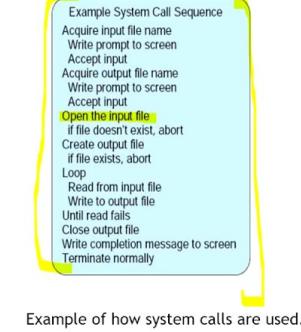




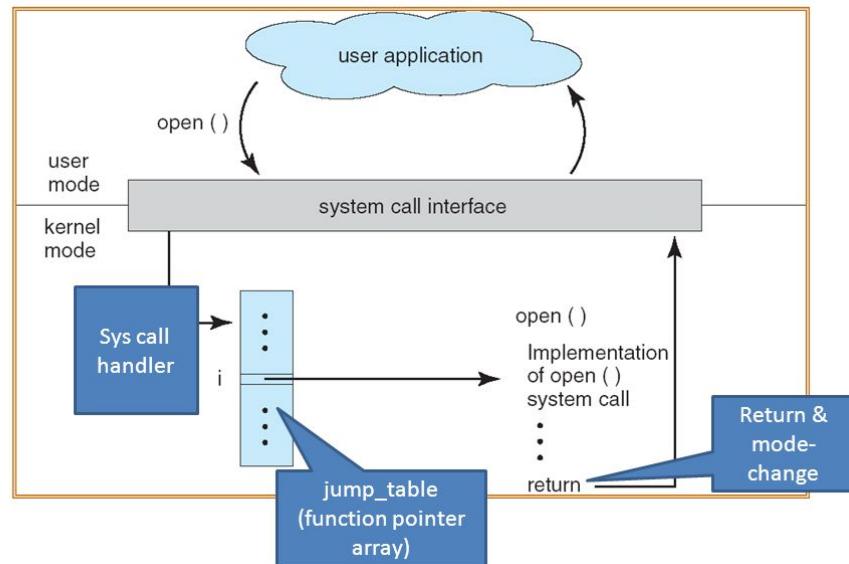
System Call



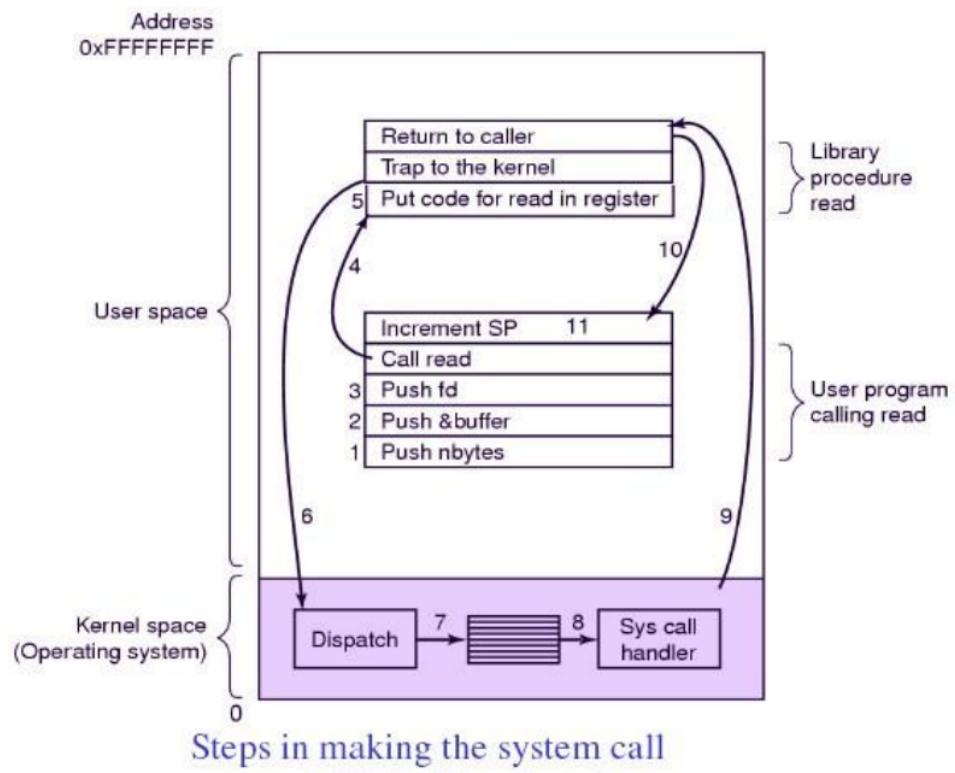
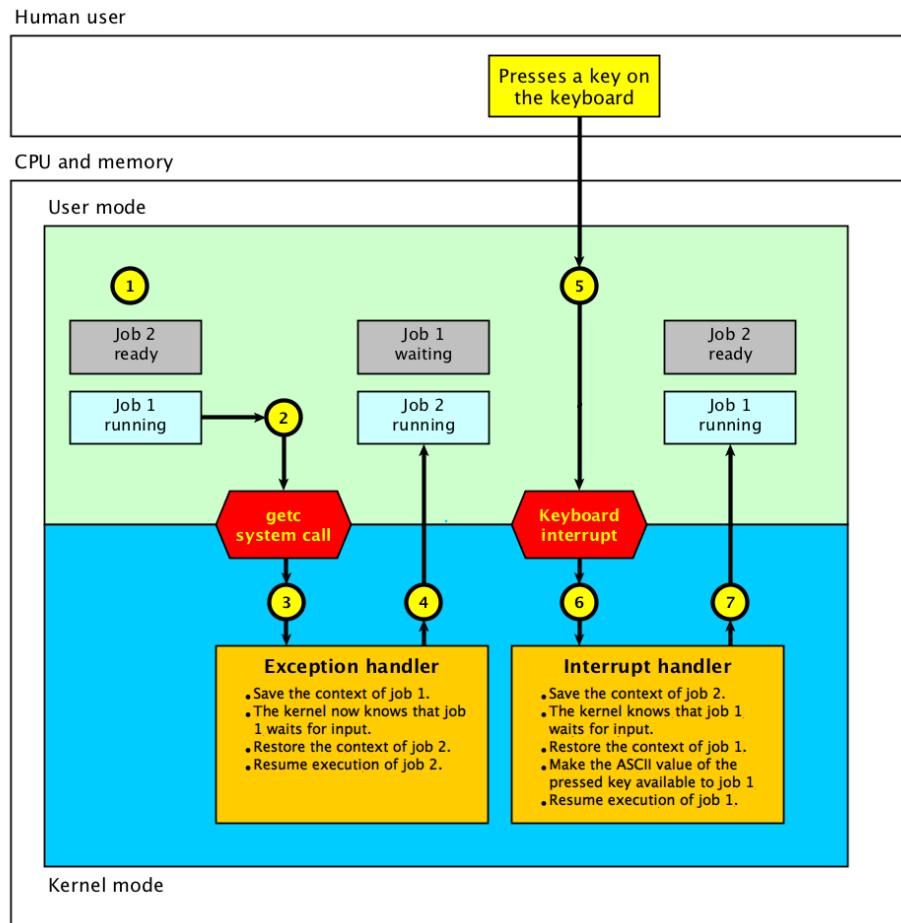
SYSTEM CALLS



API – System Call – OS Relationship



Time



API VERSUS SYSTEM CALL

API	SYSTEM CALL
A set of protocols, routines, functions that programmers use to develop software to facilitate interaction between distinct systems	A programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on
Helps to exchange data between various systems, devices and applications	Allows a program to access services from the kernel of the operating system

Visit www.PEDIAA.com

SYSTEM CALL VERSUS LIBRARY CALL

SYSTEM CALL	LIBRARY CALL
A request by the program to the kernel to enter kernel mode to access a resource	A request made by the program to access a function defined in a programming library
The mode changes from user mode to kernel mode	There is no mode switching
Not portable	Portable
Execute slower than library calls	Execute faster than system calls
System calls have more privileges than library calls	Library calls have less privileges than system calls
fork() and exec() are some examples for system calls	fopen(), fclose(), scanf() and printf() are some examples for library calls

Visit www.PEDIAA.com

OPERATING SYSTEM VERSUS APPLICATION SOFTWARE

OPERATING SYSTEM	APPLICATION SOFTWARE
A system software that manages computer hardware and software resources and provides common services for computer programs	A software designed to perform a group of coordinated functions, tasks or activities for the benefit of the user
Works as the interface between the user and hardware, performs process management, memory management, task scheduling, hardware device controlling and many more	Performs a single specific task
Developed using C, C++, Assembly languages	Developed using Java, Visual Basic, C, C++
Boots up when the user switches on the computer and runs till he switches off the machine	Runs only when the user requests to run the application
Necessary for the proper functioning of the computer	Cannot be installed without an operating system
Ex: Windows, Unix, Linux, DOS	Ex: Word, Spreadsheet, Presentation, Multimedia tools, Database Management Systems

Visit www.PEDIAA.com

Type of System Call



Types of System Calls

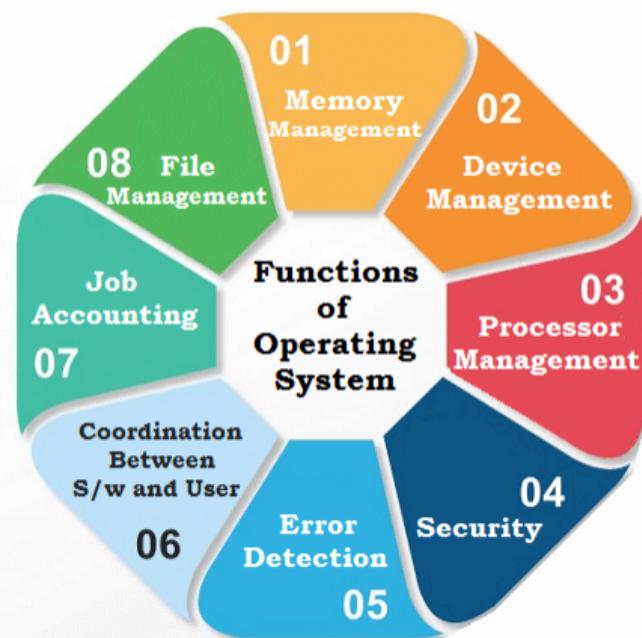
- s Process control**
 - q Load, execute, create process, wait, etc.
 - q Differs between single-tasking and multi-tasking.
- s File management**
 - q Create/delete file, open/close, read/write, etc.
- s Device management**
 - q Read, write, reposition, attach/detach device, etc.
- s Information maintenance.**
 - q Get time/date/process/file, set time/date/process/file, etc.
- s Communications**
 - q Send/receive messages , create/delete communication, etc.
 - q Two models for IPC (interprocess communication):
messages-passing and shared-memory.



EXAMPLES OF WINDOWS AND UNIX SYSTEM CALLS

	Windows	Unix
Process Control	CreateProcess() ExitProcess() WaitForSingleObject()	fork() exit() wait()
File Manipulation	CreateFile() ReadFile() WriteFile() CloseHandle()	open() read() write() close()
Device Manipulation	SetConsoleMode() ReadConsole() WriteConsole()	ioctl() read() write()
Information Maintenance	GetCurrentProcessID() SetTimer() Sleep()	getpid() alarm() sleep()
Communication	CreatePipe() CreateFileMapping() MapViewOfFile()	pipe() shmget() mmap()
Protection	SetFileSecurity() InitializeSecurityDescriptor() SetSecurityDescriptorGroup()	chmod() umask() chown()

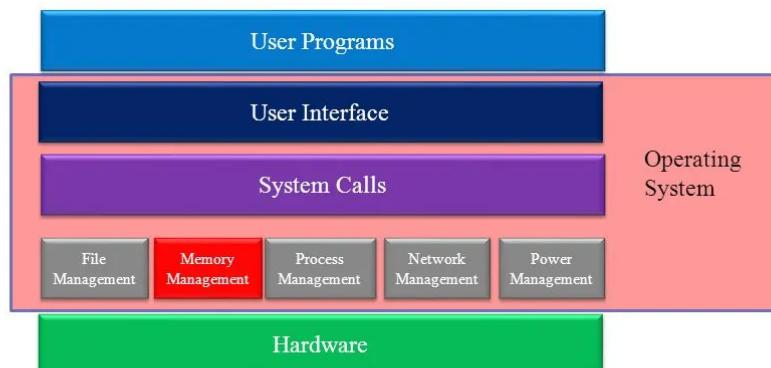
Functions of OS

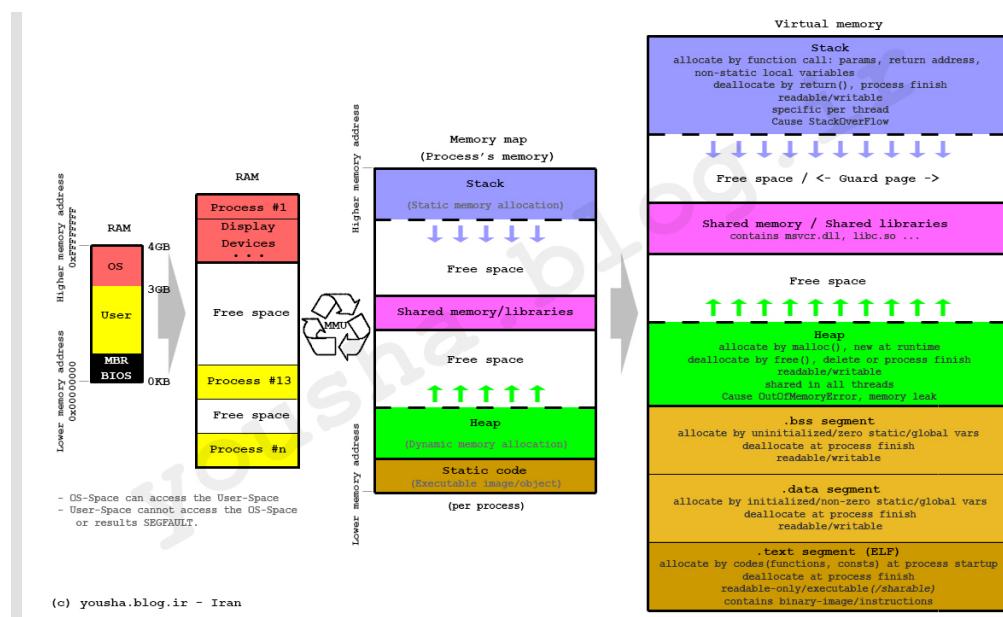
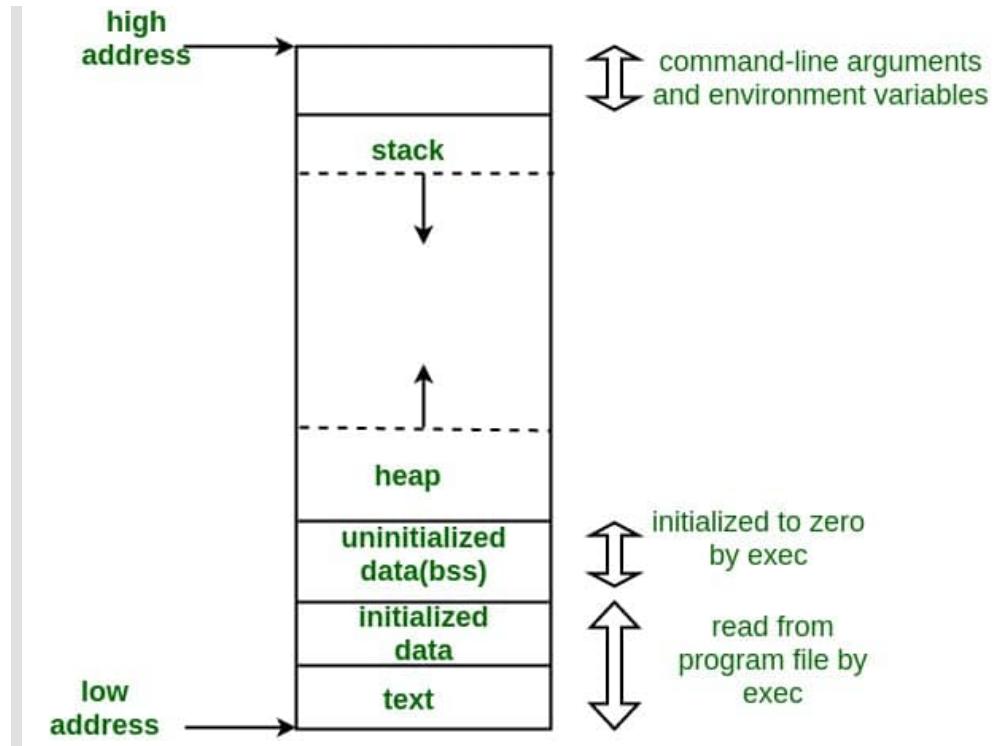


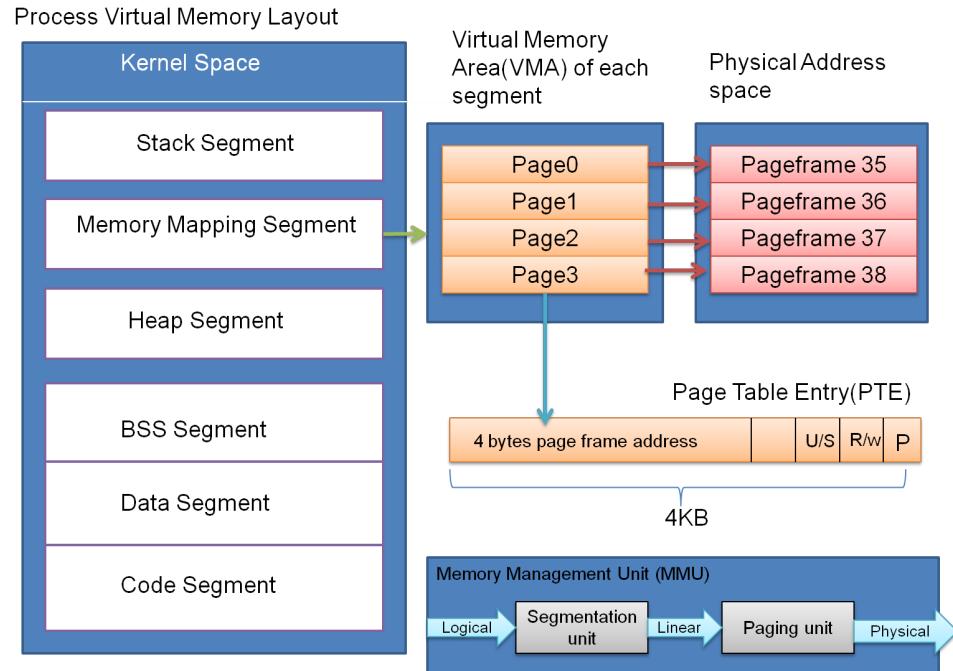
- Memory Management



Memory Management in OS

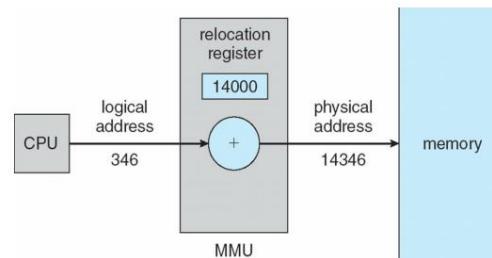






Memory-Management Unit (MMU)

- Hardware device that maps virtual to physical address
- In MMU scheme, the value in the relocation register is added to every address generated by a user process at the time it is sent to memory
- The user program deals with *logical addresses*; it never sees the *real* physical addresses

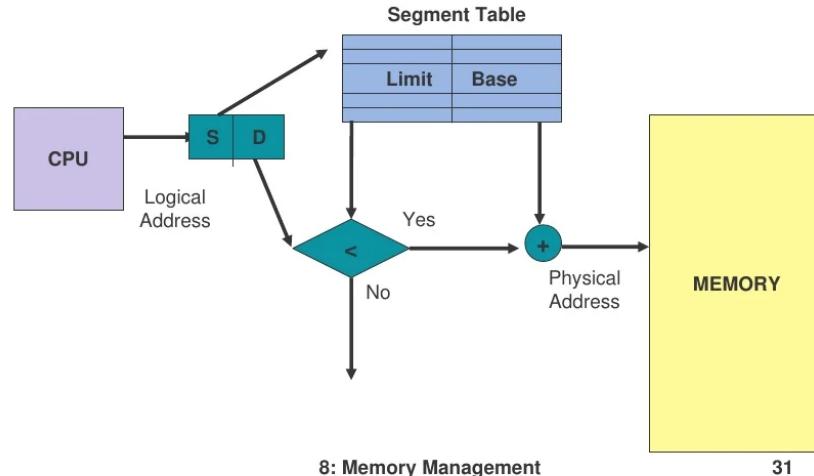


Dynamic relocation using a relocation register



MEMORY MANAGEMENT Segmentation

HARDWARE -- Must map a dyad (segment / offset) into one-dimensional address.



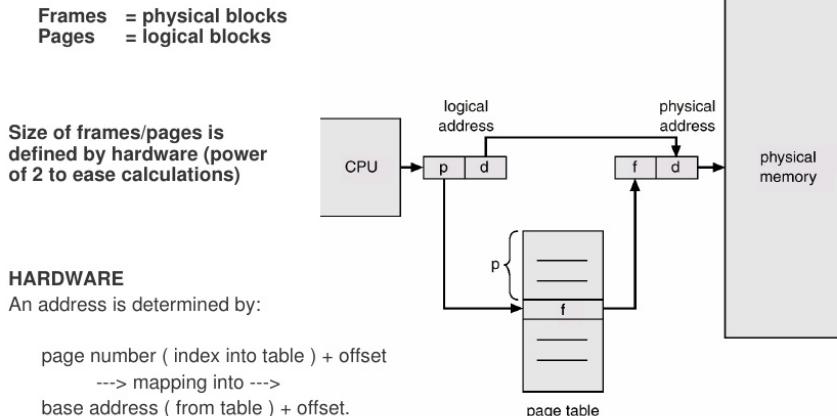
8: Memory Management

31

MEMORY MANAGEMENT

PAGING

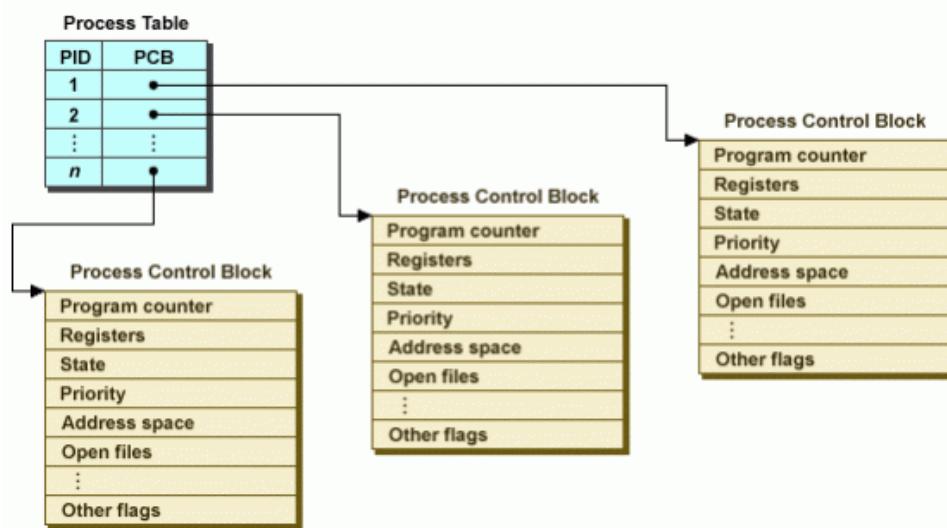
Permits a program's memory to be physically noncontiguous so it can be allocated from wherever available. This avoids fragmentation and compaction.



8: Memory Management

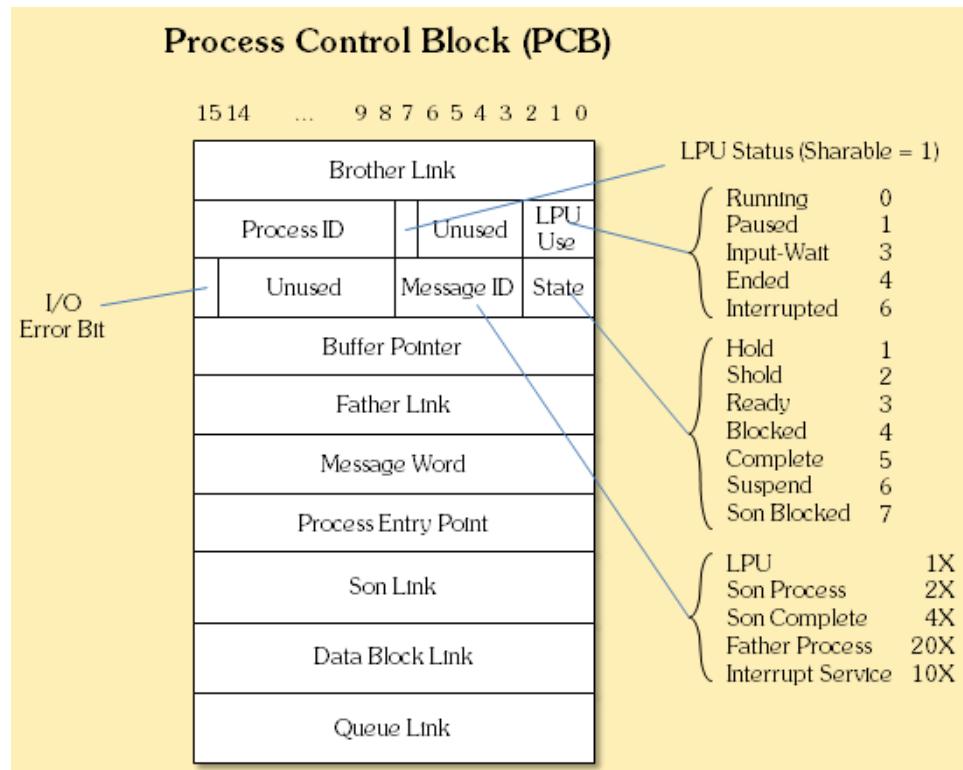
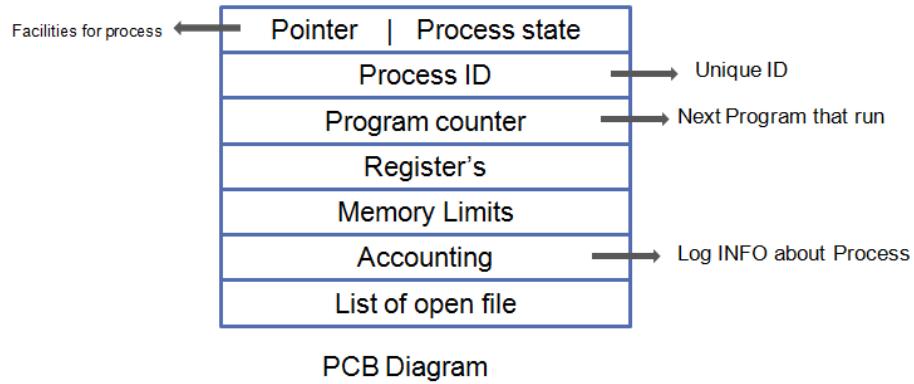
22

- Process Management



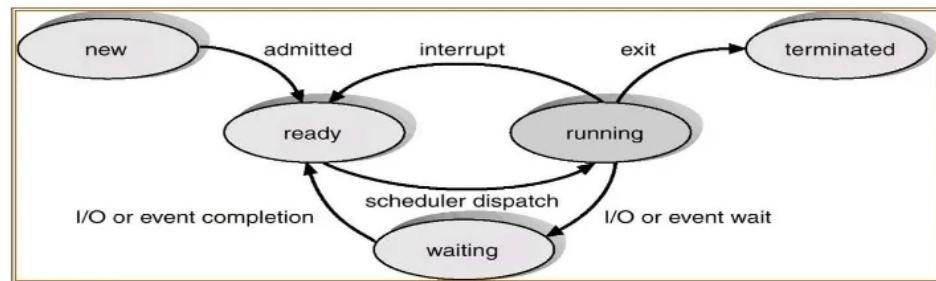
PCB(Process Control Block)

PROCESS CONTROL BLOCK (PCB)



Process Switch and State Flow

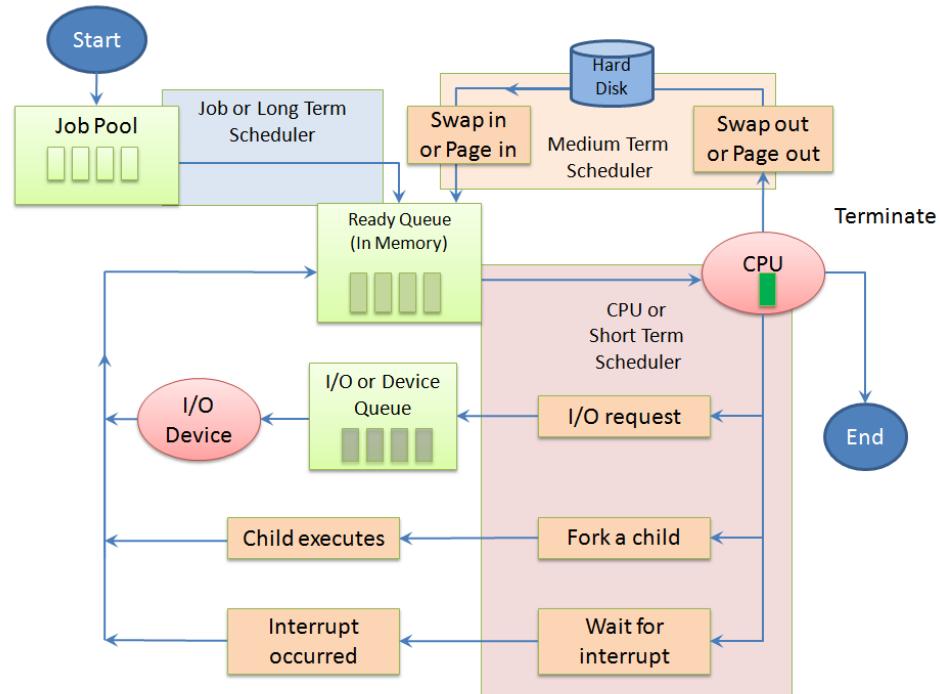
Process Management

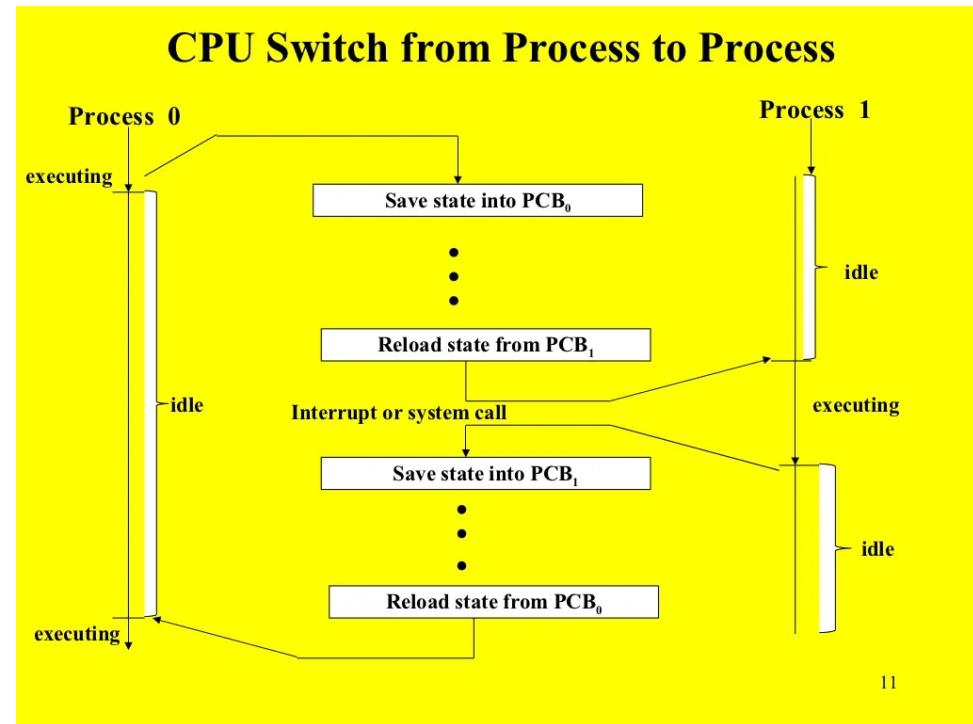


Process States

- New- The process is being created.
- Running- Instructions are being executed.
- Waiting- The process is waiting for some event to occur.
- Ready- The process is waiting to be assigned to a processor.
- Terminated- The process has finished execution.

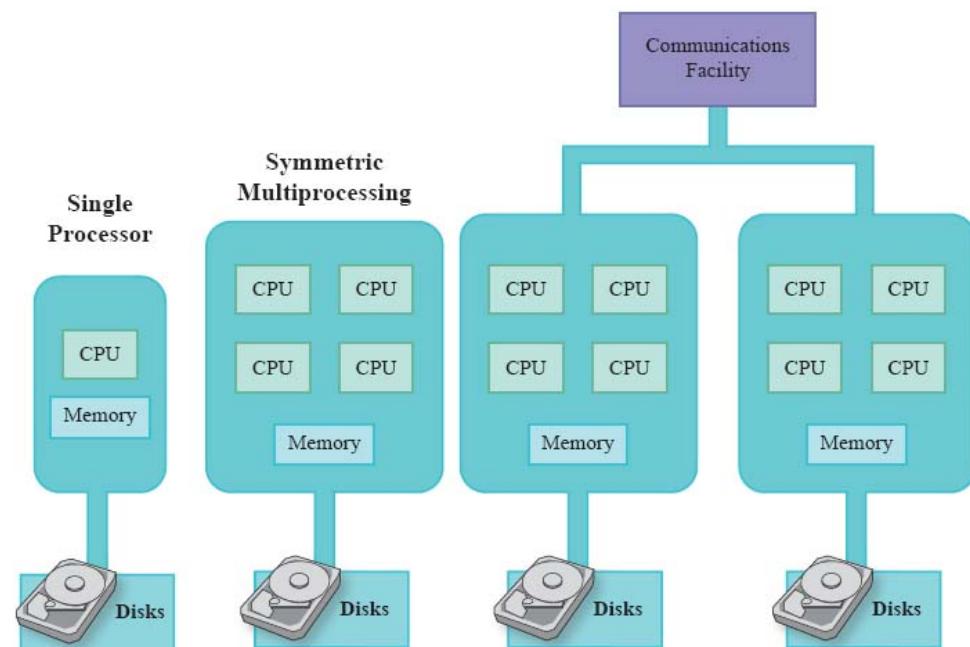
Process Control Block- contains all information associated with a specific process like process state, program counter, CPU registers and info regarding CPU scheduling algorithms, memory, I/O and accounting.





SP vs MP vs Cluster

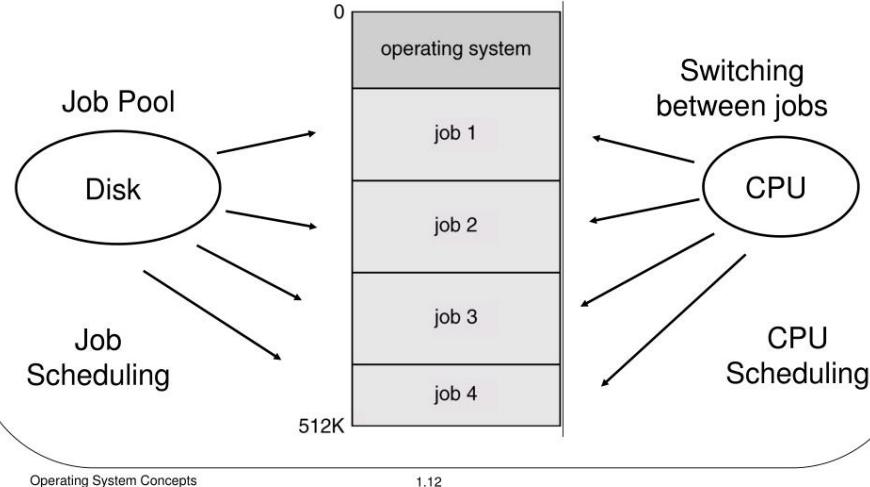
Clustered Symmetric Processing



Multiprogramming

Multiprogramming Batch Systems

Multiprogramming: several jobs are kept in main memory at the same time, and the CPU is multiplexed among them which requires memory management and protection.



MULTIPROGRAMMING VERSUS MULTITASKING

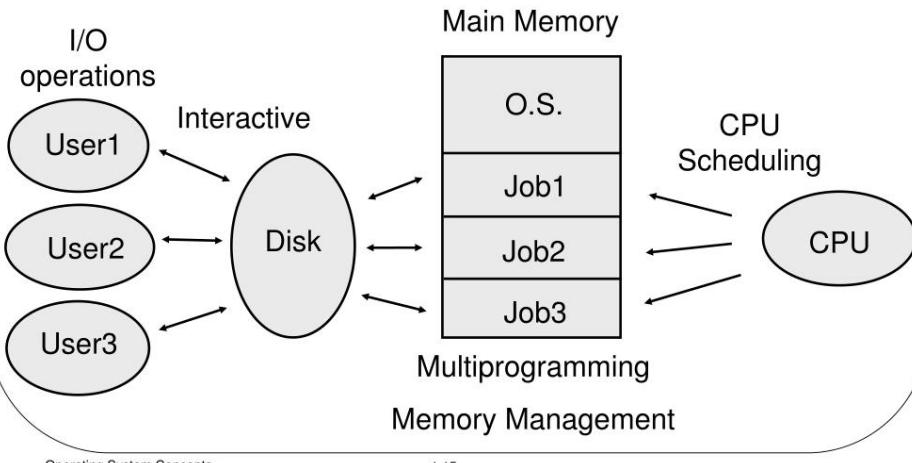
Multiprogramming	Multitasking
In multiprogramming, multiple processes run concurrently at the same time on a single processor.	Multitasking is when more than one task is executed at a single time utilizing multiple CPUs
It is based on the concept of context switching.	It is based on the concept of time sharing.
Multiple programs reside in the main memory simultaneously to improve CPU utilization so that CPU doesn't sit idle for a long time.	It enables execution of multiple tasks and processes at the same time to increase CPU performance.
It utilizes single CPU for execution of processes.	It utilizes multiple CPUs for task allocation.
It takes more time to execute the processes.	It takes less time to execute the tasks or processes.
The idea is to reduce the CPU idle time for as long as possible.	The idea is to allow multiple processes to run simultaneously via time sharing.

DB Difference Between.net

Time Sharing

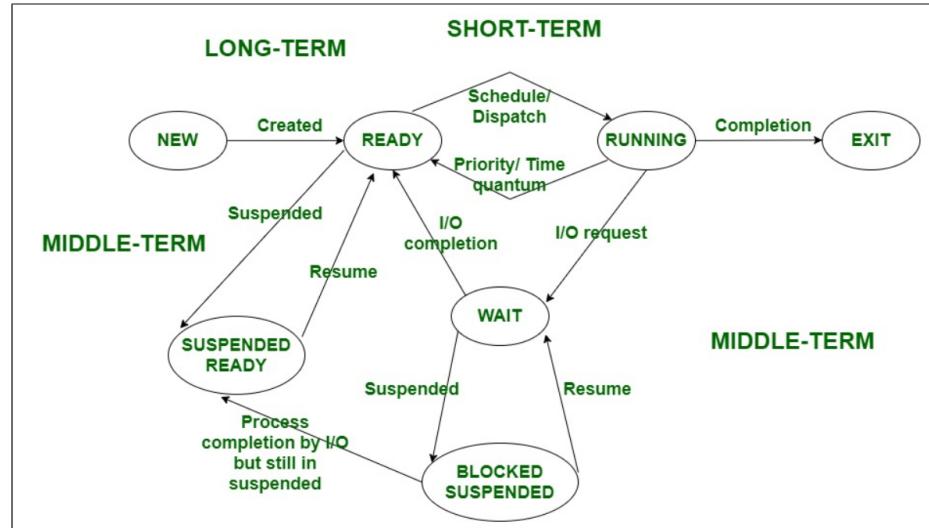
Time-Sharing Systems – Interactive Computing

A time-sharing system uses CPU scheduling and multiprogramming to provide each user with a small portion of a time-shared computer.

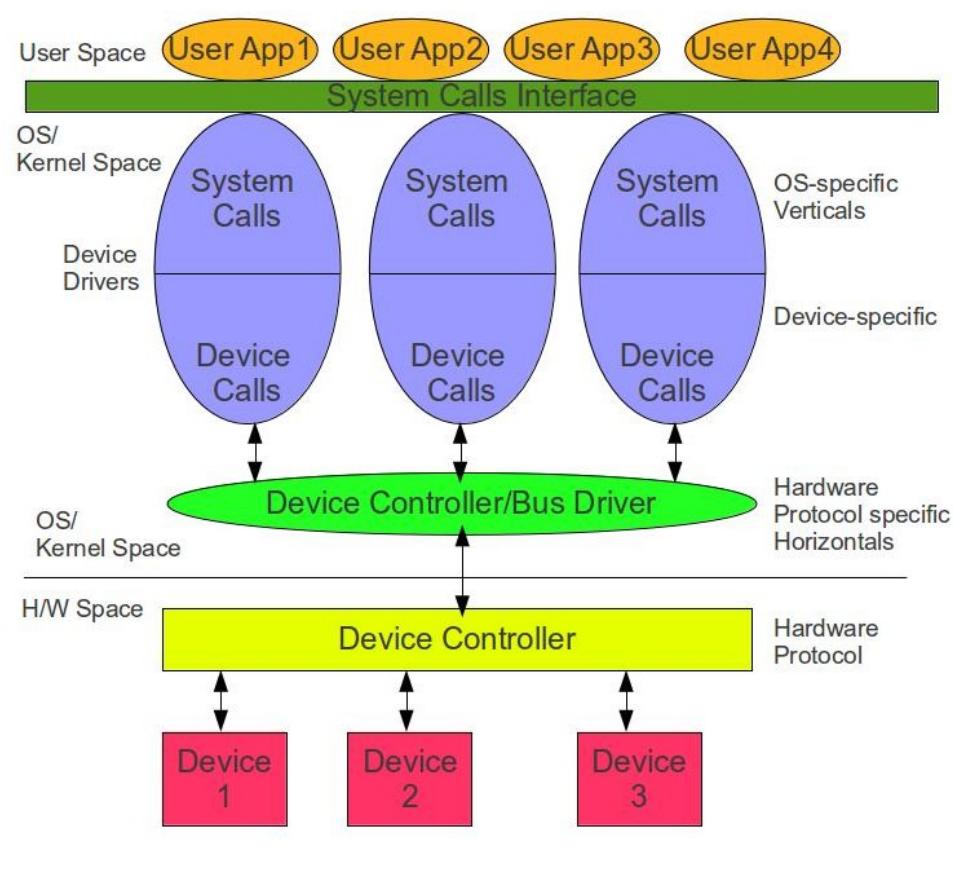


Operating System Concepts

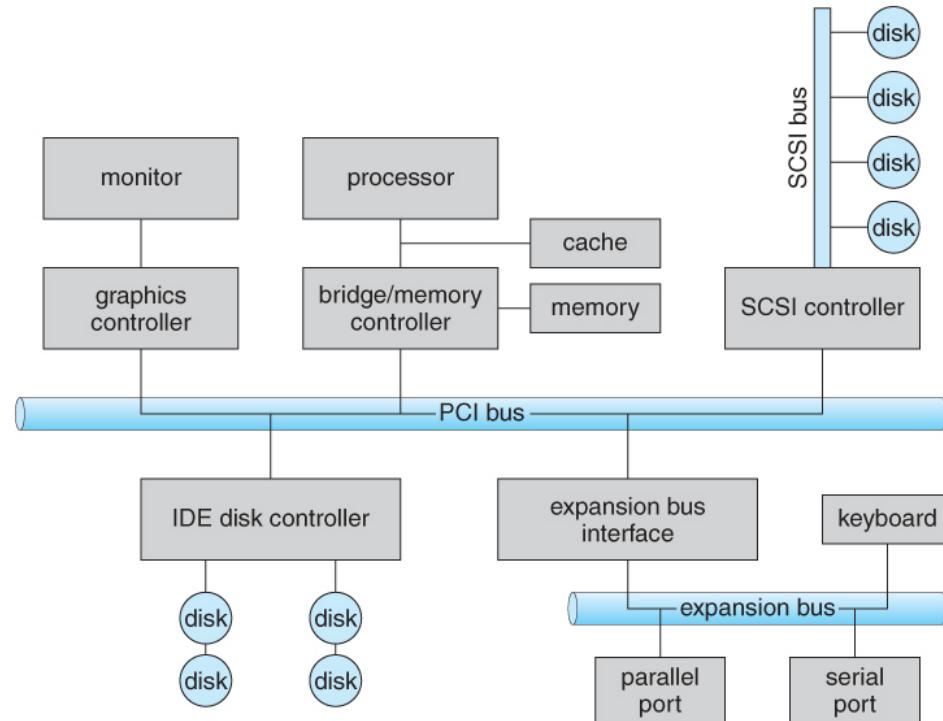
1.15



- Device Management



- I/O Management



- OS Security

Precautions for OS Security

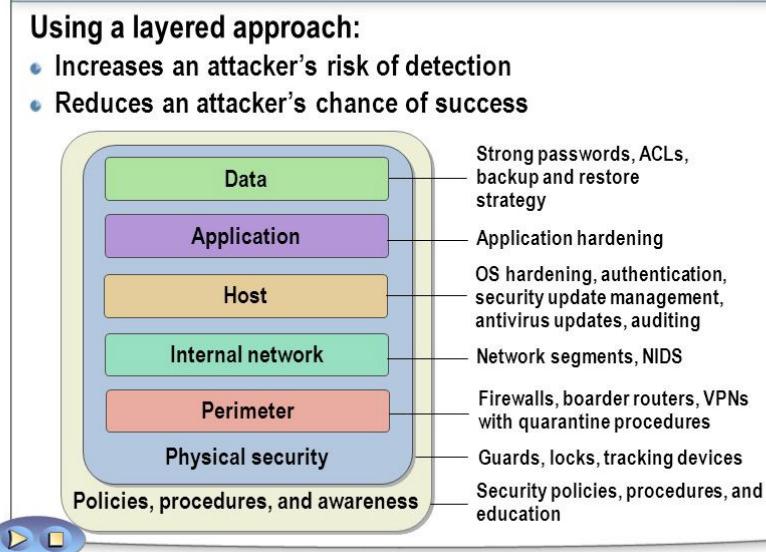
To Secure Our System and to avoid the Security breaches we must take some precautions in our OS.

So we have to ensure of the Shown Security Components:-

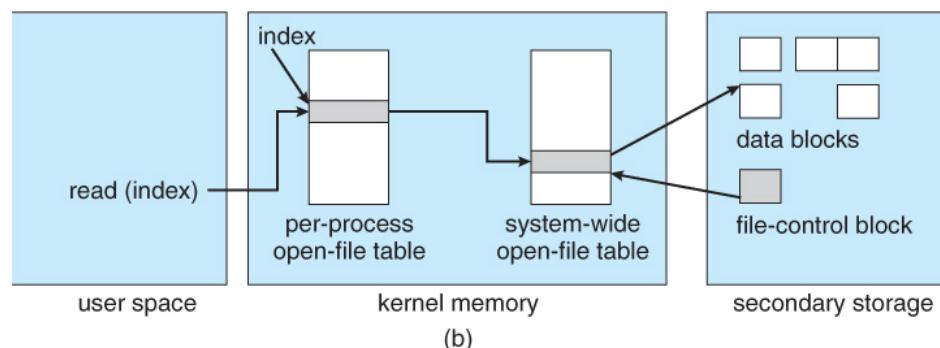
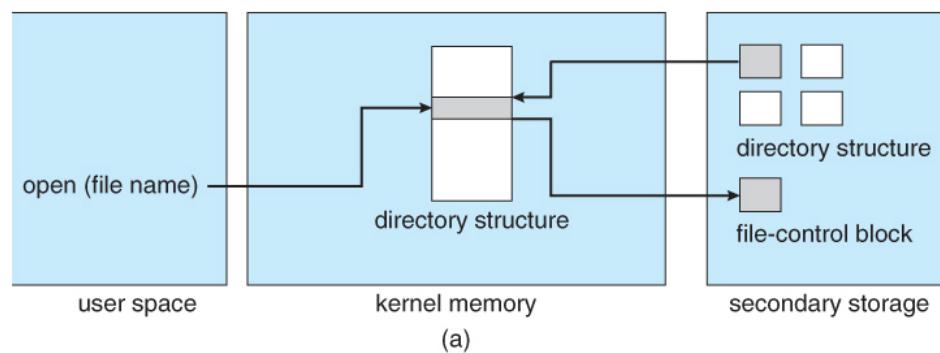
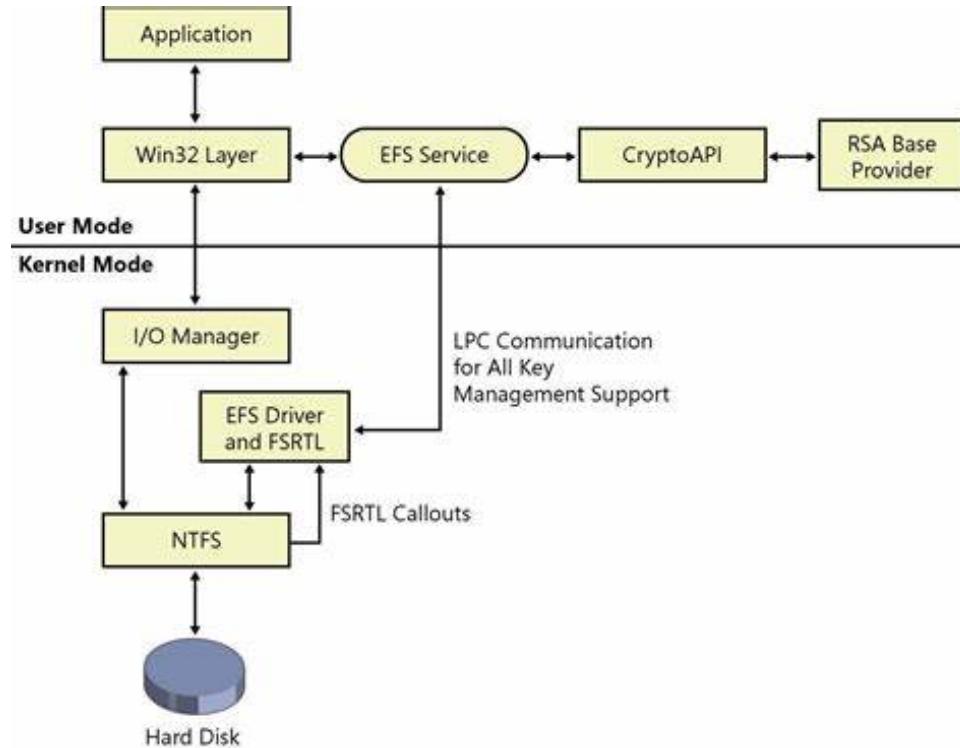
- Bios Security
- User Accounts Security
- Data Security
- Antivirus Security
- Firewall



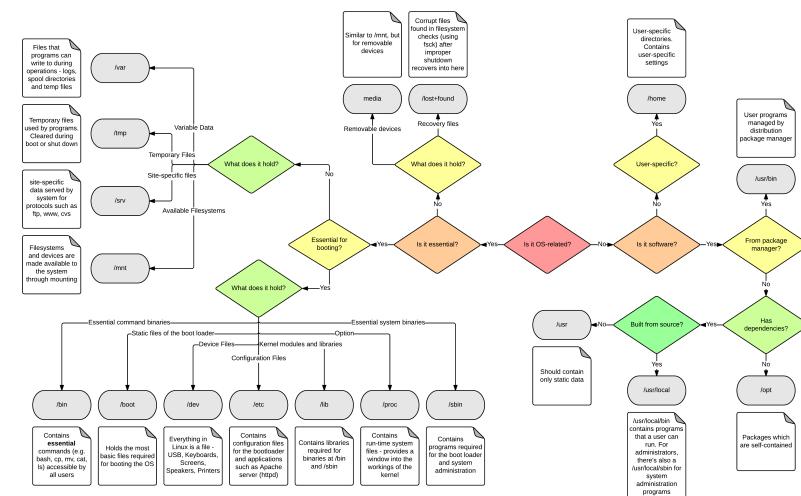
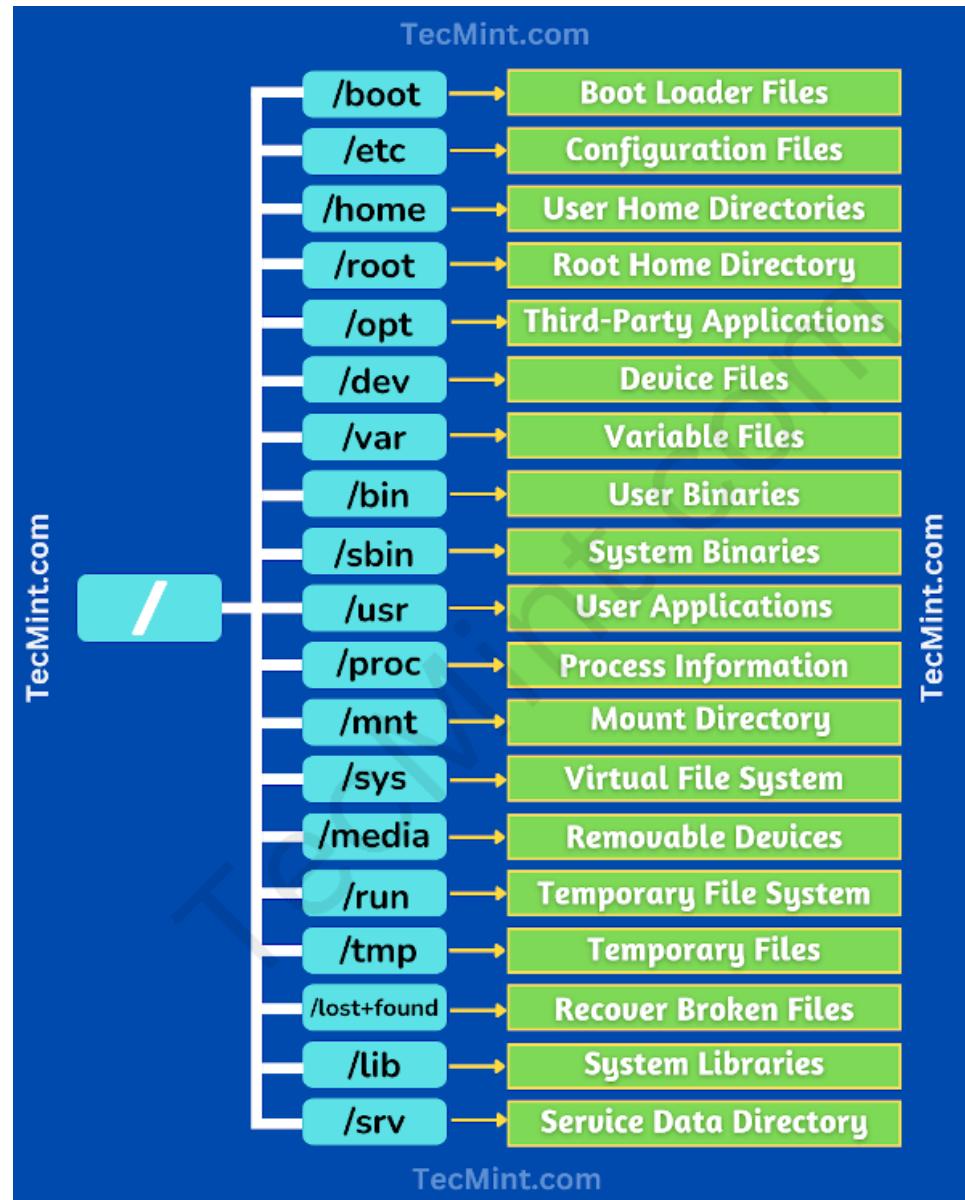
Understanding Defense-in-Depth



- File Management

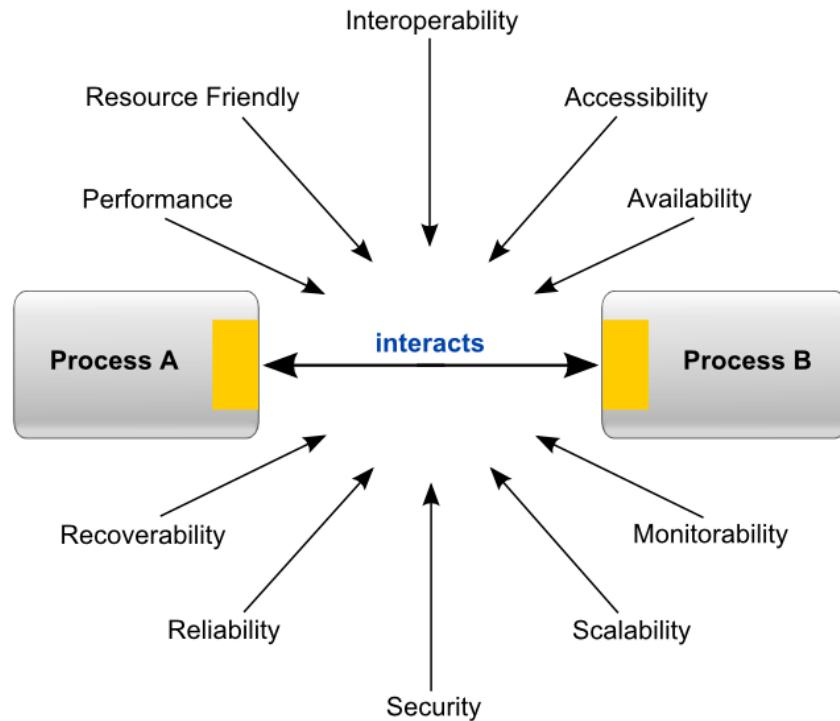


File type	Usual extension	Function
Executable	exe,com,bin	Read to run machine language program
Object	obj,o	Compiled,machine language not linked
Source code	C,java,pas,asm,a	Source code in various languages
Batch	bat,sh	Commands to the command interpreter
Text	txt,doc	Textual data,documents
Word processor	Wp,tex,rrf,doc	Various word processor formats
Archive	arc,zip,tar	Related files grouped into one file compressed



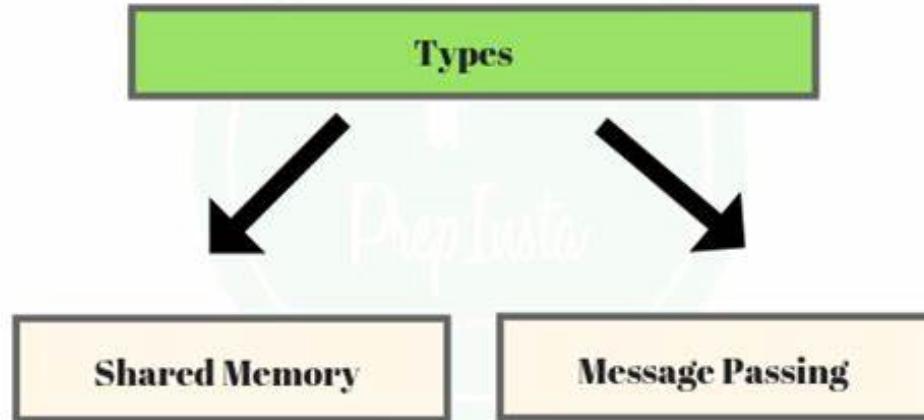
Process Communication

Interprocess Communication



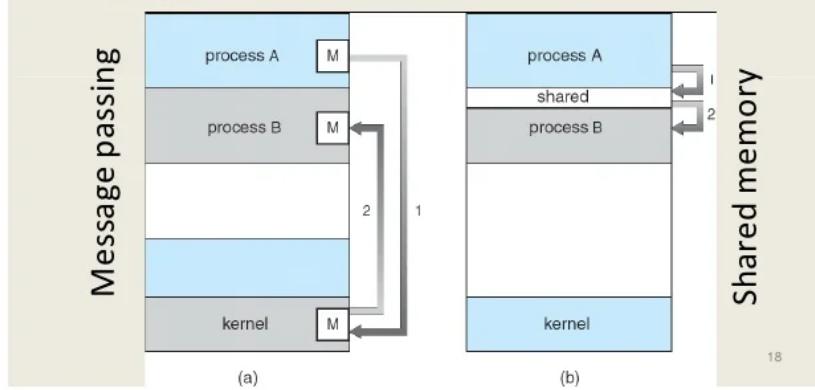


Inter-Process Communication Types



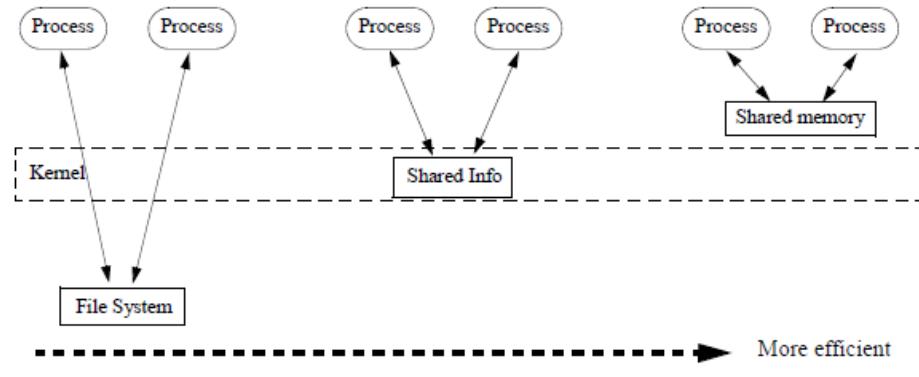
4. Interprocess Communication Contd...

- Two fundamental models of Interprocess communication
- **Shared memory**
 - a region of memory that is shared by cooperating processes is established then exchange information takes place by reading and writing data to the shared region
- **Message passing**
 - communication takes place by means of messages exchanged between the cooperating processes



IPC Mechanisms

IPC classified by implementation mechanisms.

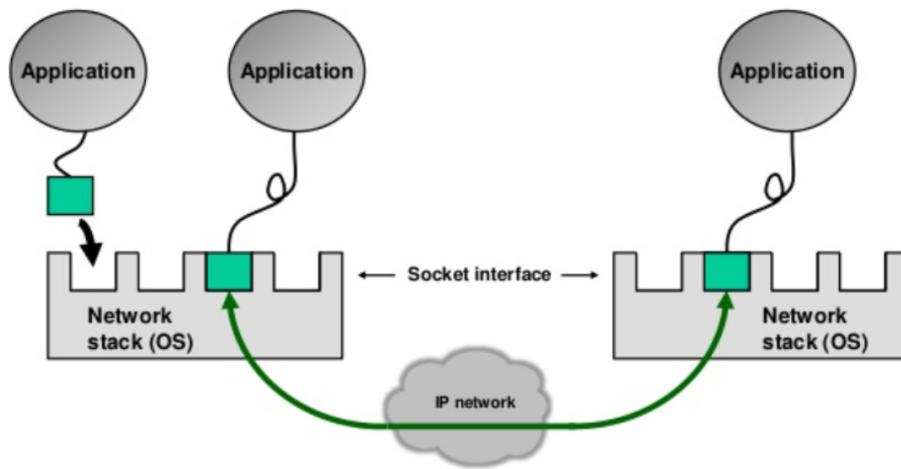


Examples:

files

semaphore, socket,
pipe, message queue,
signal

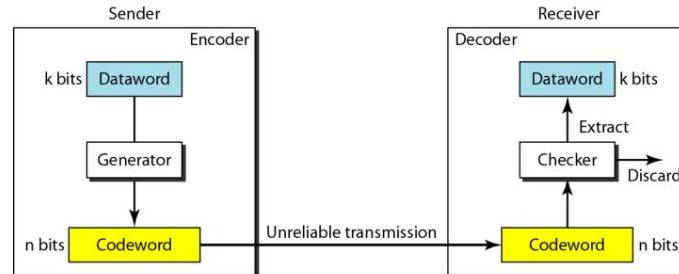
shared memory



Error Detection and Correction

Error Detection

- A receiver can detect a change if the original codeword if
 - The receiver has a list of valid codewords, and
 - The original codeword has changed to an invalid one.



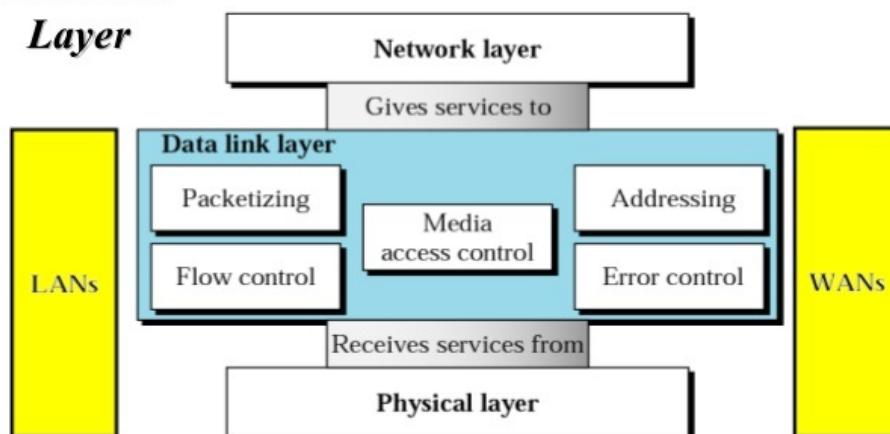
10.12

Er. M.S.Kuthar

Error Detection and Correction

Data can be corrupted during transmission.
Some applications require that errors be detected and corrected.

Data Link Layer



Error Detection & Correction

■ Error Detection

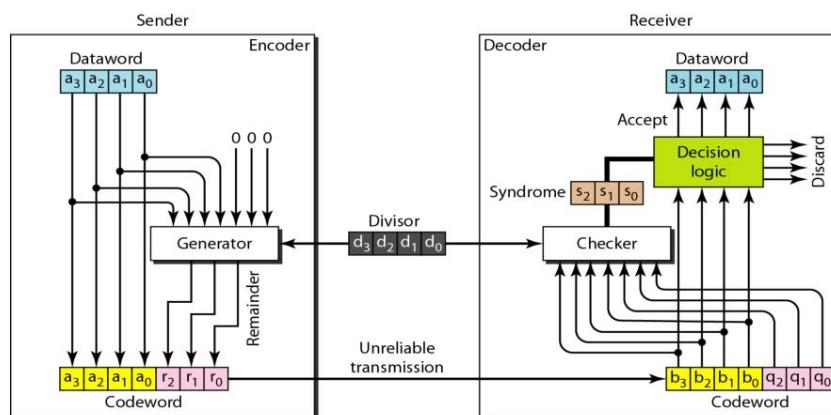
- Check if any error has occurred
- Don't care the number of errors
- Don't care the positions of errors

■ Error Correction

- Need to know the number of errors
- Need to know the positions of errors
- More complex
- The number of errors and the size of the message are important factors in error correction

Error Detection & Correction

Figure 1.6 CRC encoder and decoder



1. Resource Allocation

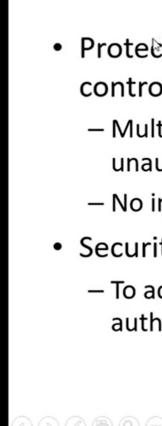
- In case of multiple users accessing same resource
- In case of multiple processes accessing same resource
- Example:
 - Multiple users may be accessing same resource (say printer), then OS allocates the printer based on some algo (like FIFO for ex)
 - CPU Scheduling algos (FIFO, SJF etc)



S

2. Accounting

- Statistics related to the resource usage by the users – how much resource each user consumes, what all resources a user uses etc.
- Can be used for billing purposes
- Can also help in reconfiguring system to improve computing services



S

3. Protection and Security

- Protection – access to system should be controlled
 - Multiuser systems should not allow an unauthorized user to access content
 - No interference between 2 processes
- Security – from external world
 - To access a system, there must be some kind of authorization (like a password)



S

Programming Language - HTML

Special HTML character entities

&	'	()	*	+	,
<small>\kern .1em \&</small>	<small>\kern .1em '</small>	<small>\kern .1em (</small>	<small>\kern .1em)</small>	<small>\kern .1em *</small>	<small>\kern .1em +</small>	<small>\kern .1em ,</small>
.	/	:	;	<	=	>
<small>\kern .1em .</small>	<small>\kern .1em /</small>	<small>\kern .1em :</small>	<small>\kern .1em ;</small>	<small>\kern .1em <</small>	<small>\kern .1em =</small>	<small>\kern .1em ></small>
?	@	[\]	^	—
<small>\kern .1em ?</small>	<small>\kern .1em @</small>	<small>\kern .1em [</small>	<small>\kern .1em \backslash</small>	<small>\kern .1em]</small>	<small>\kern .1em ^</small>	<small>\kern .1em —</small>
`	{	 	}			
<small>\kern .1em `</small>	<small>\kern .1em {</small>	<small>\kern .1em </small>	<small>\kern .1em }</small>			
£	¤	¥	 	§	..	©
<small>\kern .1em £</small>	<small>\kern .1em ¤</small>	<small>\kern .1em ¥</small>	<small>\kern .1em </small>	<small>\kern .1em §</small>	<small>\kern .1em ..</small>	<small>\kern .1em ©</small>

Latex Language

LATEX Mathematical Symbols

The more unusual symbols are not defined in base L^AT_EX (NFSS) and require `\usepackage{amssymb}`

1 Greek and Hebrew letters

α	<code>\alpha</code>	κ	<code>\kappa</code>	ψ	<code>\psi</code>	\digamma	<code>\digamma</code>	Δ	<code>\Delta</code>	Θ	<code>\Theta</code>
β	<code>\beta</code>	λ	<code>\lambda</code>	ρ	<code>\rho</code>	\varkappa	<code>\varkappa</code>	Γ	<code>\Gamma</code>	Υ	<code>\Upsilon</code>
χ	<code>\chi</code>	μ	<code>\mu</code>	σ	<code>\sigma</code>	φ	<code>\varphi</code>	Λ	<code>\Lambda</code>	Ξ	<code>\Xi</code>
δ	<code>\delta</code>	ν	<code>\nu</code>	τ	<code>\tau</code>	φ	<code>\varphi</code>	Ω	<code>\Omega</code>	Ω	<code>\Omega</code>
ϵ	<code>\epsilon</code>	ω	<code>\omega</code>	θ	<code>\theta</code>	ϖ	<code>\varpi</code>	Φ	<code>\Phi</code>	\aleph	<code>\aleph</code>
η	<code>\eta</code>	ω	<code>\omega</code>	υ	<code>\upsilon</code>	ϱ	<code>\varrho</code>	Π	<code>\Pi</code>	\beth	<code>\beth</code>
γ	<code>\gamma</code>	ϕ	<code>\phi</code>	ξ	<code>\xi</code>	ς	<code>\varsigma</code>	Ψ	<code>\Psi</code>	\daleth	<code>\daleth</code>
ι	<code>\iota</code>	π	<code>\pi</code>	ζ	<code>\zeta</code>	ϑ	<code>\vartheta</code>	Σ	<code>\Sigma</code>	\gimel	<code>\gimel</code>

2 L^AT_EX math constructs

$\frac{abc}{xyz}$	<code>\frac{abc}{xyz}</code>	\overline{abc}	<code>\overline{abc}</code>	\overrightarrow{abc}	<code>\overrightarrow{abc}</code>
f'	<code>f'</code>	\underline{abc}	<code>\underline{abc}</code>	\overleftarrow{abc}	<code>\overleftarrow{abc}</code>
\sqrt{abc}	<code>\sqrt{abc}</code>	\widehat{abc}	<code>\widehat{abc}</code>	\overbrace{abc}	<code>\overbrace{abc}</code>
$\sqrt[n]{abc}$	<code>\sqrt[n]{abc}</code>	\widetilde{abc}	<code>\widetilde{abc}</code>	\underbrace{abc}	<code>\underbrace{abc}</code>

3 Delimiters

$ $	$ $	$\{$	$\{$	\lfloor	\lfloor	$/$	$/$	\uparrow	\uparrow	\uparrow	\uparrow
\mid	\mid	$\}$	$\}$	\rfloor	\rfloor	\backslash	\backslash	\uparrow	\uparrow	\uparrow	\uparrow
$\ $	$\ $	\langle	\langle	\lceil	\lceil	\lfloor	\lfloor	\downarrow	\downarrow	\downarrow	\downarrow
\parallel	\parallel	\rangle	\rangle	\rceil	\rceil	\rfloor	\rfloor	\downarrow	\downarrow	\downarrow	\downarrow

Use the pair `\left{s1` and `\right{s2}` to match height of delimiters s_1 and s_2 to the height of their contents, e.g.,
 $\left\lfloor \left. expr \right\rfloor \right\rfloor$ $\left\lfloor \left. \left\lfloor \left. expr \right\rfloor \right\rfloor \right\rfloor$ $\left\lfloor \left. \left\lfloor \left. \left. expr \right\rfloor \right\rfloor \right\rfloor \right\rfloor$

4 Variable-sized symbols (displayed formulae show larger version)

\sum	<code>\sum</code>	\int	<code>\int</code>	\bigcup	<code>\bigcup</code>	\oplus	<code>\oplus</code>	\bigvee	<code>\bigvee</code>
\prod	<code>\prod</code>	\oint	<code>\oint</code>	\bigcap	<code>\bigcap</code>	\otimes	<code>\otimes</code>	\bigotimes	<code>\bigotimes</code>
\coprod	<code>\coprod</code>	\iint	<code>\iint</code>	\bigcup	<code>\bigcup</code>	\odot	<code>\odot</code>	\bigodot	<code>\bigodot</code>

5 Standard Function Names

Function names should appear in Roman, not Italic, e.g.,

Correct: `\tan(at-n\pi)` $\rightarrow \tan(at - n\pi)$

Incorrect: `\tan(at-n\pi)` $\rightarrow \tan(at - n\pi)$

\arccos	<code>\arccos</code>	\sin	<code>\sin</code>	\arcsin	<code>\arcsin</code>	\tan	<code>\tan</code>	\arctan	<code>\arctan</code>	\arg	<code>\arg</code>
\cos	<code>\cos</code>	\cosh	<code>\cosh</code>	\cos	<code>\cos</code>	\log	<code>\log</code>	\cot	<code>\cot</code>	\coth	<code>\coth</code>
\csc	<code>\csc</code>	\deg	<code>\deg</code>	\sinh	<code>\sinh</code>	\lim	<code>\lim</code>	\det	<code>\det</code>	\dim	<code>\dim</code>
\exp	<code>\exp</code>	\gcd	<code>\gcd</code>	\hom	<code>\hom</code>	\log	<code>\log</code>	\inf	<code>\inf</code>	\liminf	<code>\liminf</code>
\ker	<code>\ker</code>	\lg	<code>\lg</code>	\lim	<code>\lim</code>	\sec	<code>\sec</code>	\tan	<code>\tan</code>	\max	<code>\max</code>
\limsup	<code>\limsup</code>	\ln	<code>\ln</code>	\log	<code>\log</code>	\tan	<code>\tan</code>	\sin	<code>\sin</code>	\tanh	<code>\tanh</code>
\min	<code>\min</code>	\Pr	<code>\Pr</code>	\sec	<code>\sec</code>	\tan	<code>\tan</code>	\arg	<code>\arg</code>	\sinh	<code>\sinh</code>
\sinh	<code>\sinh</code>	\sup	<code>\sup</code>	\tan	<code>\tan</code>						

Programming Language - JavaScript

JavaScript Reserved Words

Keywords or Reserved Words

var	delete	for	let	break
super	void	case	do	static
function	new	switch	while	interface
catch	else	if	package	finally
this	with	class	enum	default
implements	private	throw	yield	typeof
const	export	import	protected	return
true	continue	extends	in	instanceof
public	try	debugger	false	

www.geekyshows.com

bitwise operation Javascript

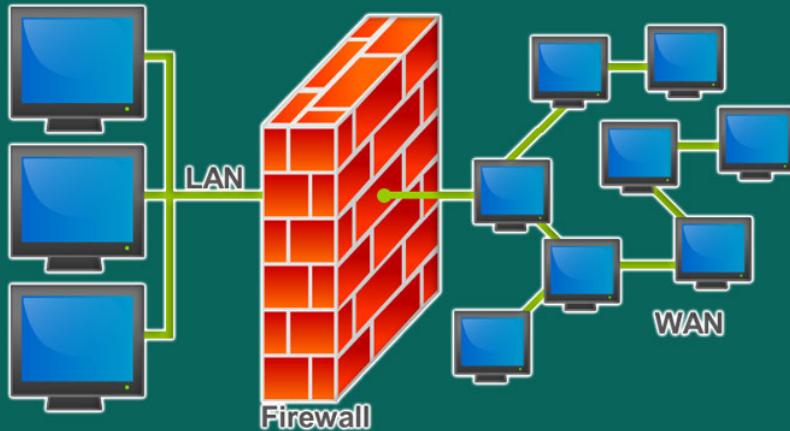
Bitwise operators in JavaScript

Syntax	Name	Type
&	Bitwise AND	binary
	Bitwise OR	binary
^	Bitwise XOR (exclusive OR)	binary
~	Bitwise NOT	unary
<<	Left shift	binary
>>	Right shift	binary
>>>	Zero-fill right shift	binary

Network

Firewall

Configure Firewall & Internet Security of the QuickBooks Desktop



What is firewall?

A firewall is nothing but a network security system that monitors and controls over all your incoming and outgoing network traffic based on advanced and a defined set of security rules.

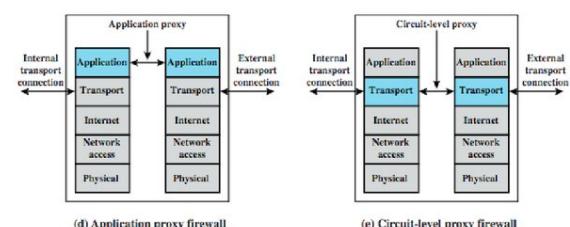
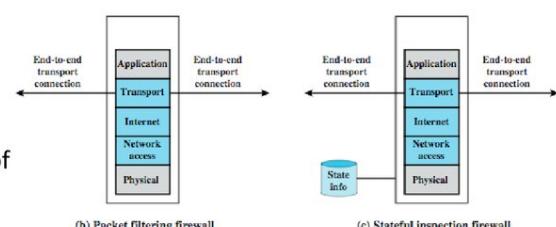
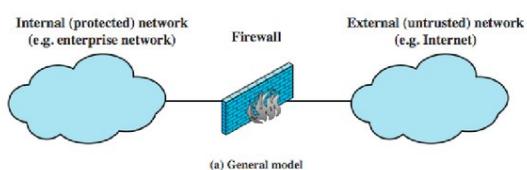
It simply prevents unauthorized access to or from a private network. Used to enhance the security of computers connected to a network, such as LAN or the Internet. Considered as an integral part of a comprehensive security framework for your network.

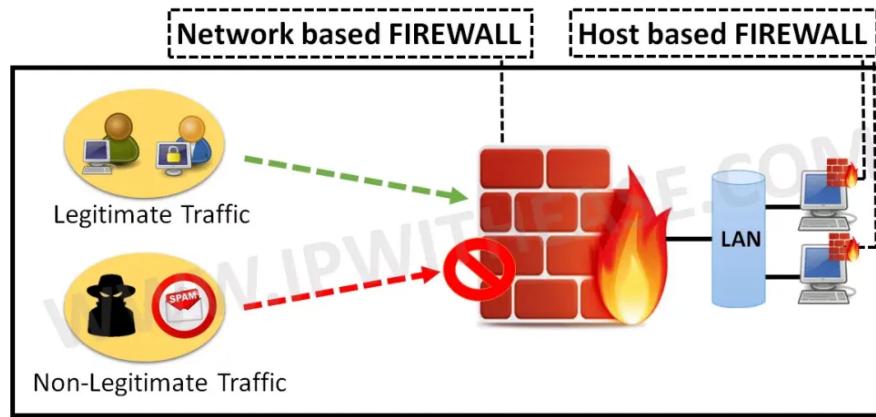


Types of Firewalls

Positive (negative) filter:
Allow (reject) packets that meet a criteria

Stateful inspection: Keeps track of TCP connections

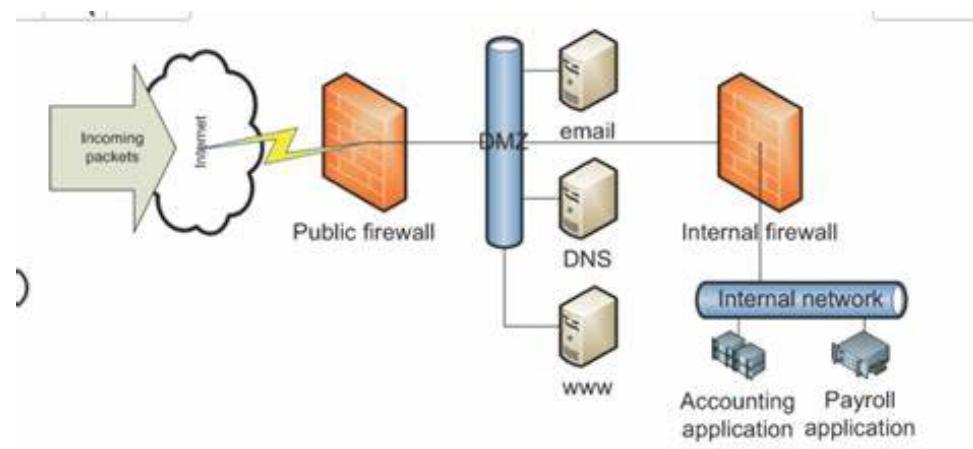


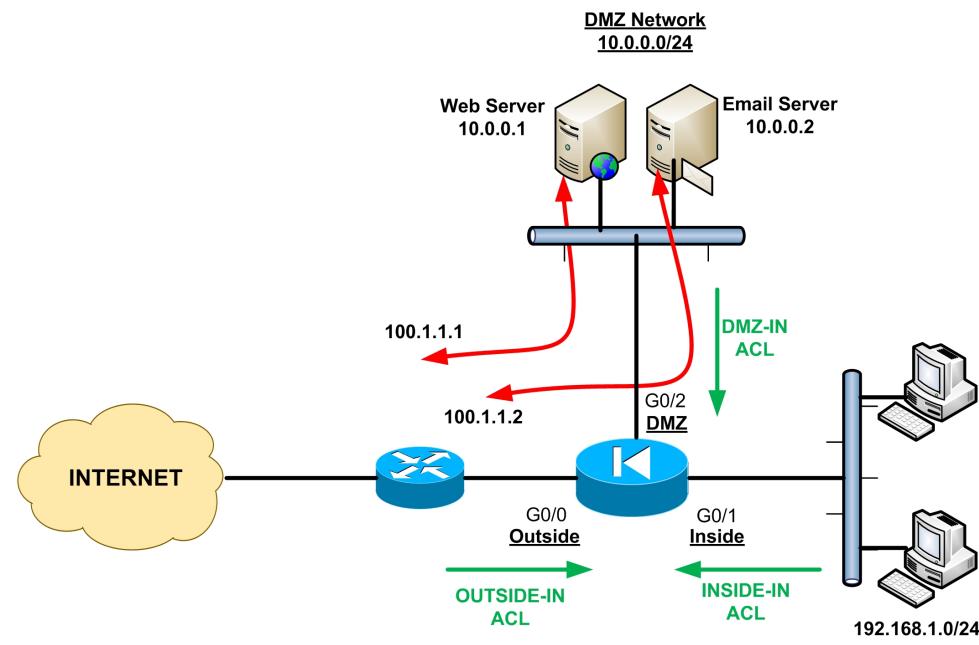


ACL (Access Control List)

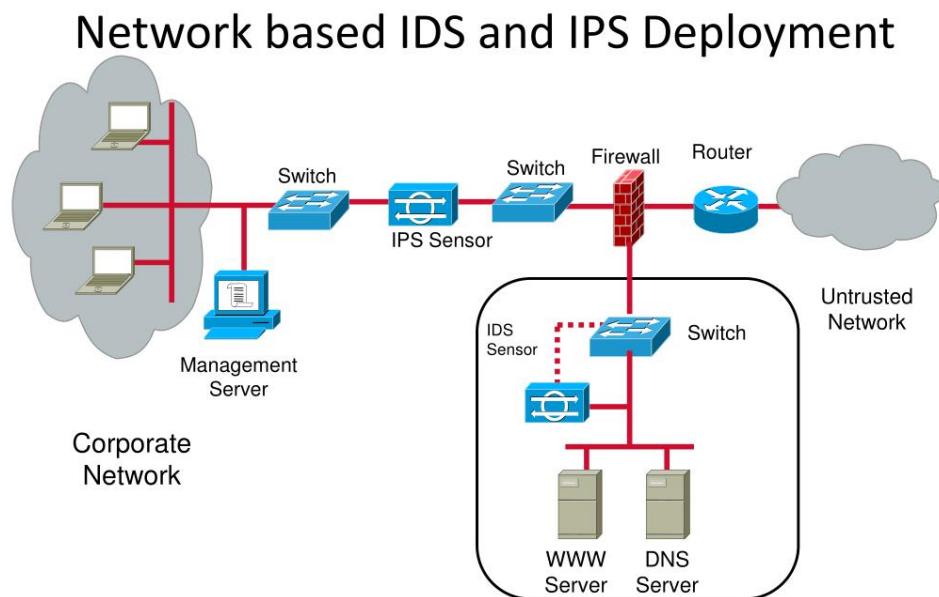
Parameter	ACL	Firewall
Asset Type	Feature on Layer 3 devices and Firewalls	Hardware or Software
Stateful/Stateless inspection	Performs stateless inspection	Performs Stateful inspection
Scope wrt OSI	Upto Layer 4	Upto Layer 7
Security	Low	High
Intrusion detection	Not possible	Possible
Target deployment	Setups requiring low level of security	Setups requiring higher level of security

DMZ





IPS/IDS

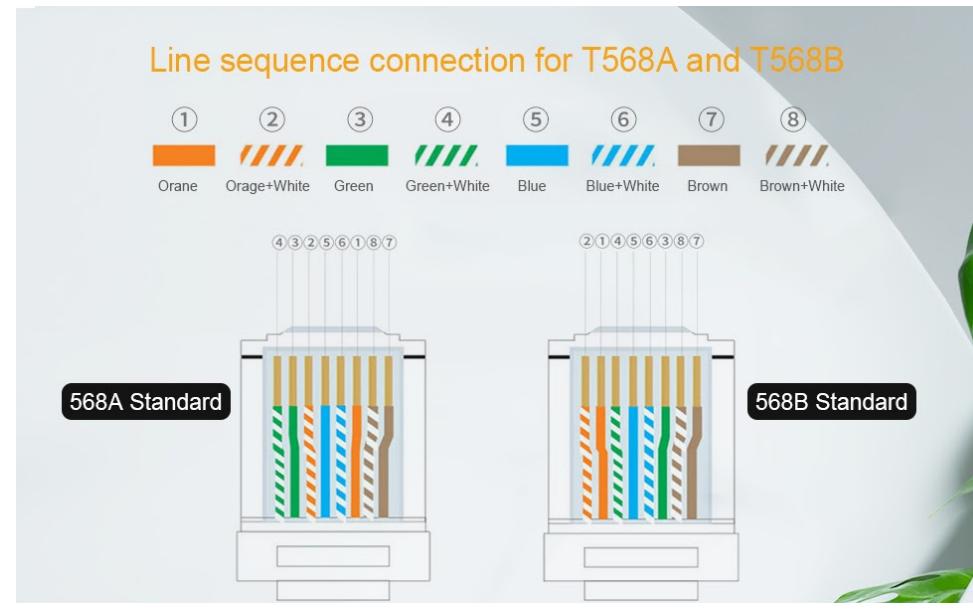


Engineering and Management of Secure Computer Networks

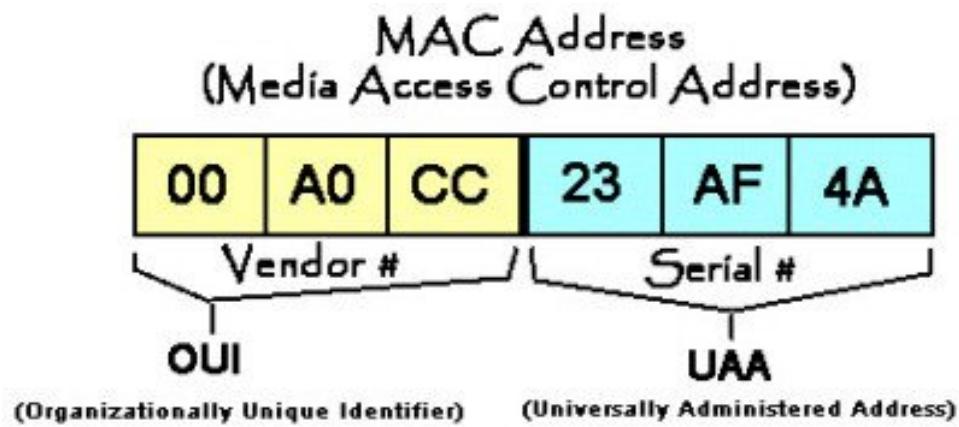
15

Cable

Category	Standard Bandwidth	Max Data Rate	Shielding
Cat5e	100MHz (up to 350)	1000Mbps	UTP or STP
Cat6	250MHz (up to 550)	1000Mbps	UTP or STP
Cat6A	500MHz (up to 550)	10Gbps	UTP or STP
Cat7	600MHz	10Gbps	Shielded only
Cat8	2000MHz	25Gbps or 40Gbps	Shielded only



MAC Address



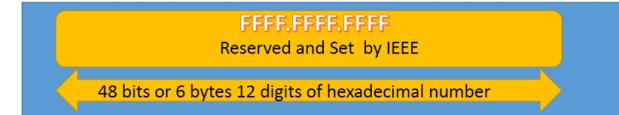
Unicast MAC address



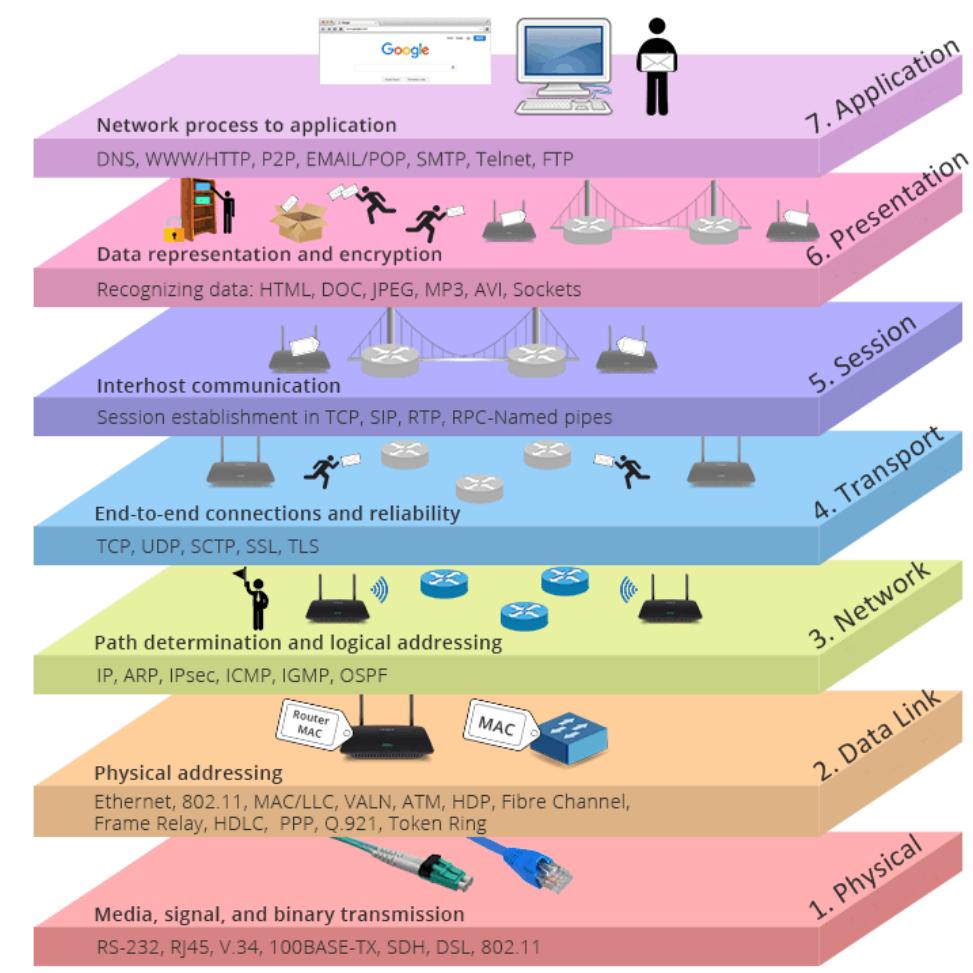
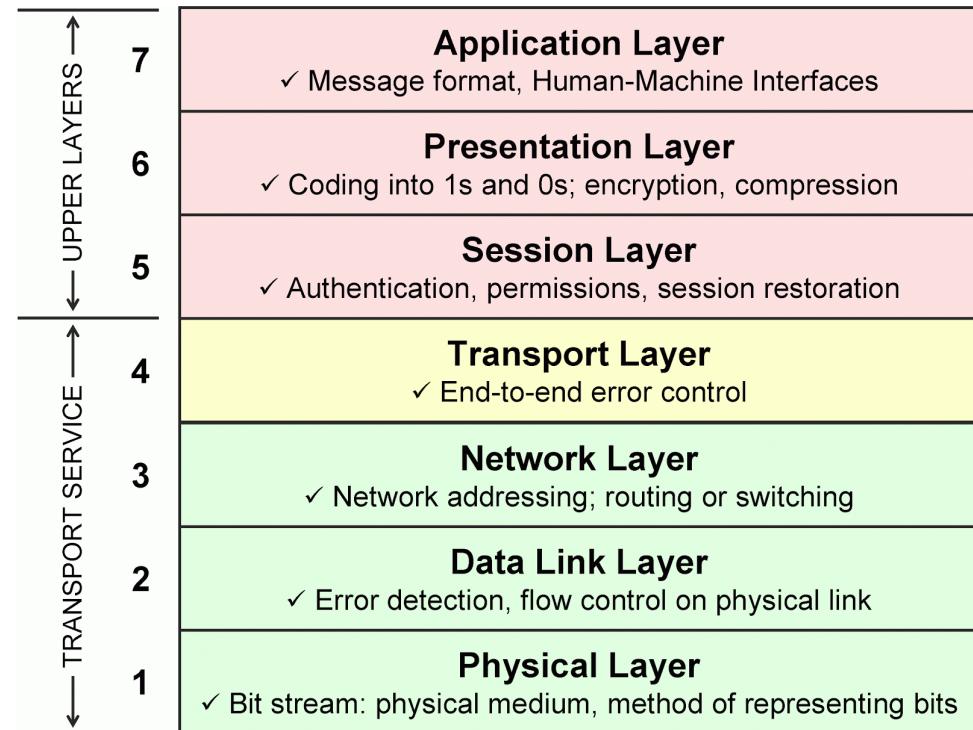
Multicast MAC address

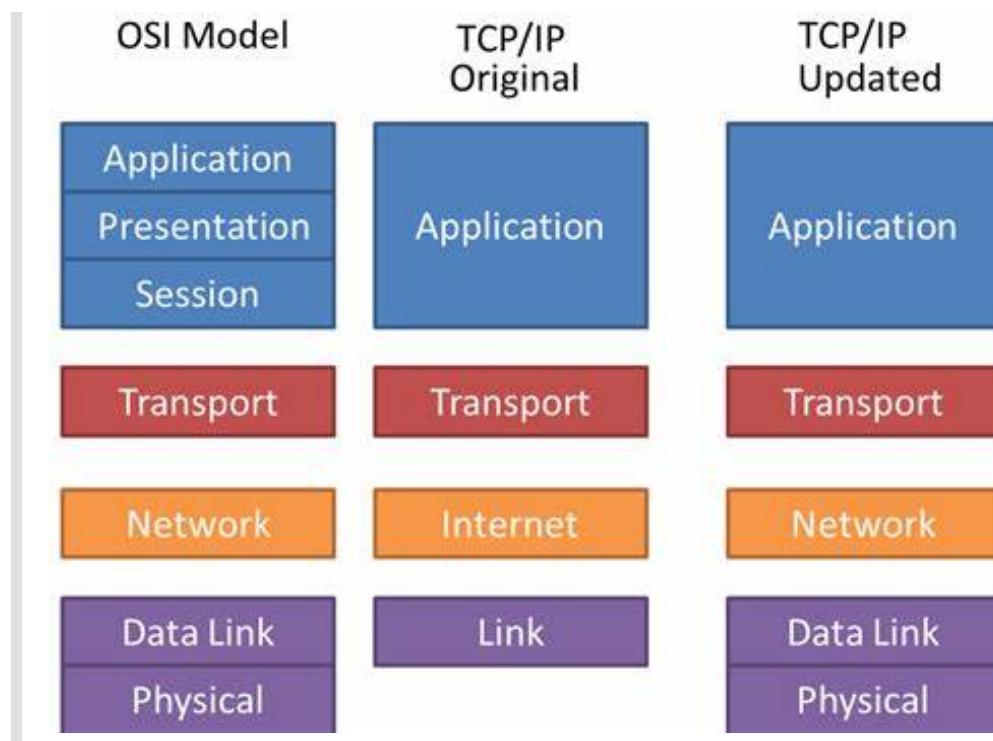


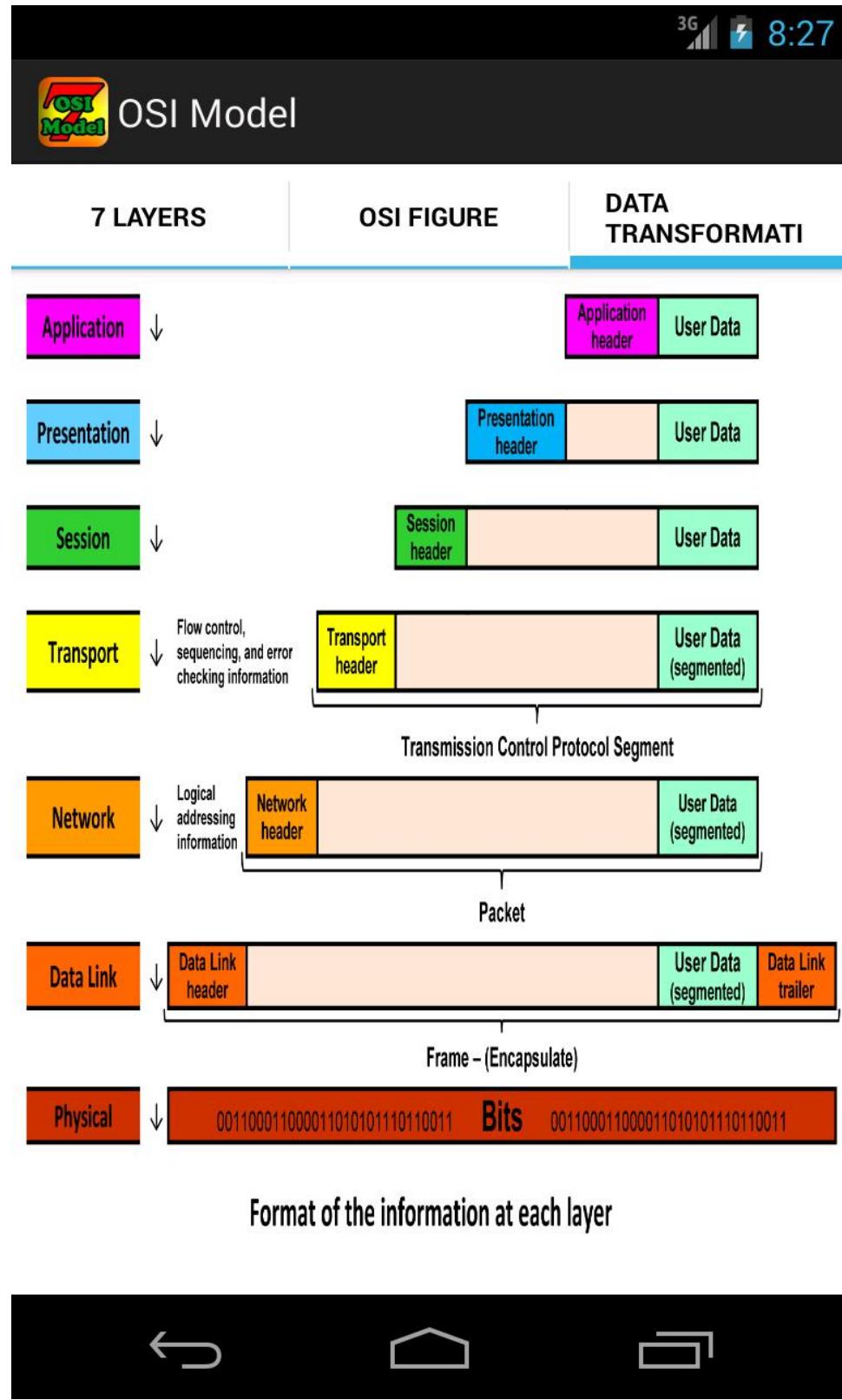
Broadcast MAC address



OSI Model

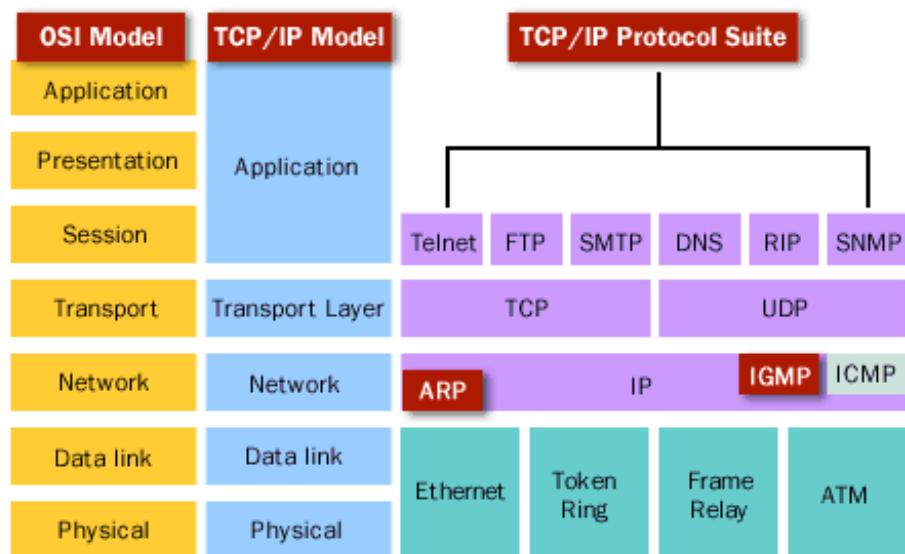
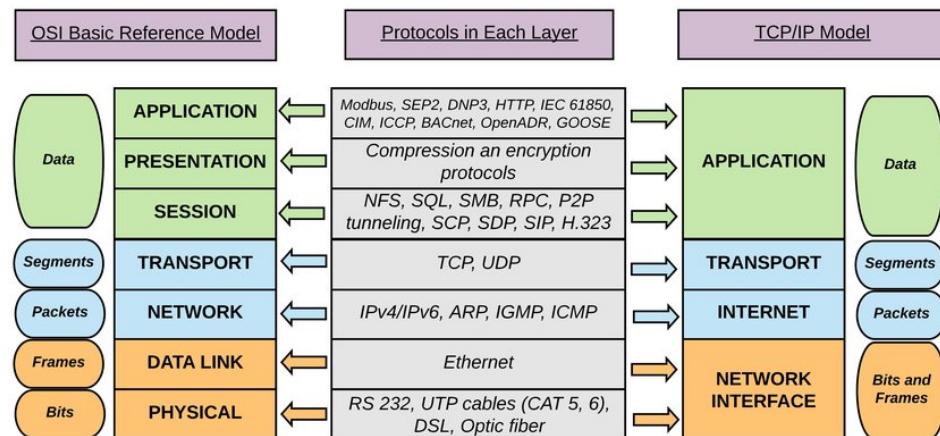
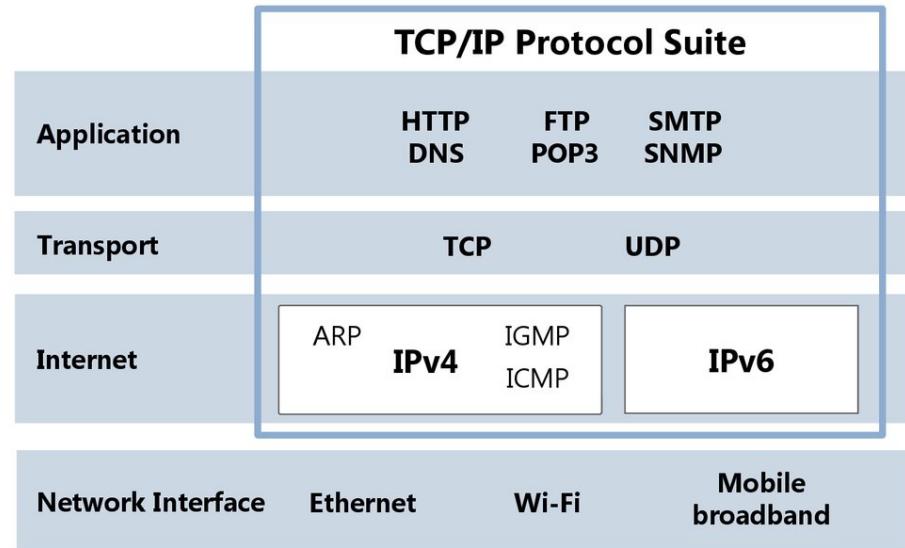




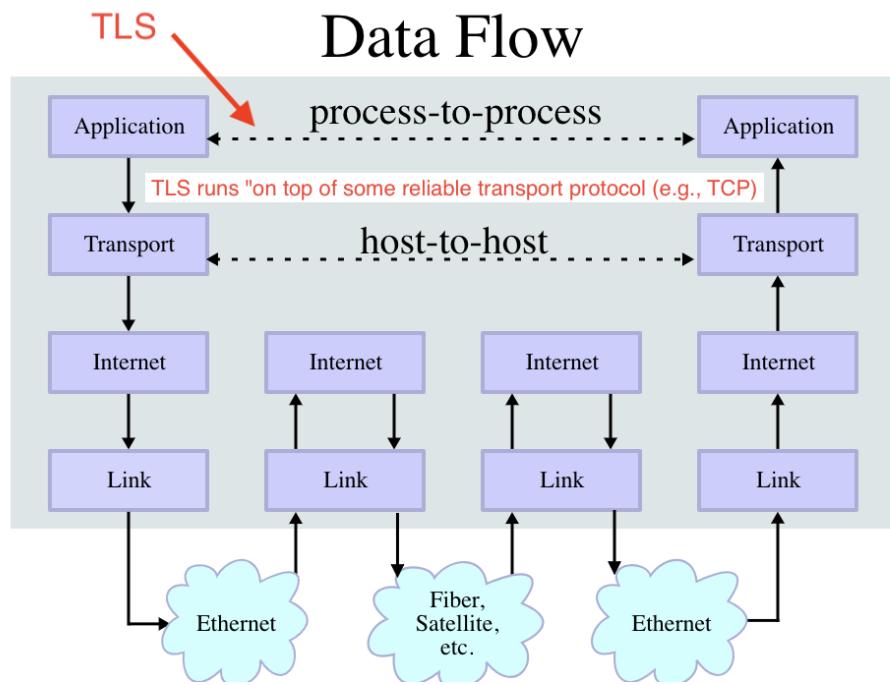
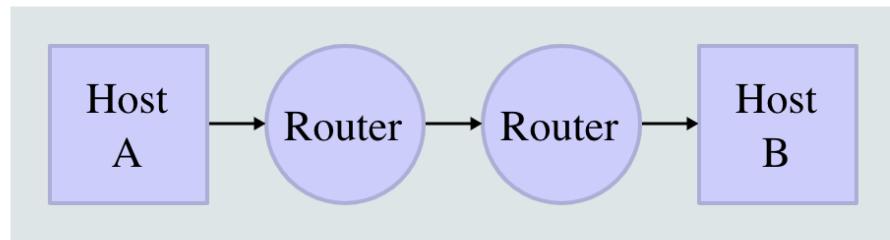


Protocol

The TCP/IP Protocol Suite

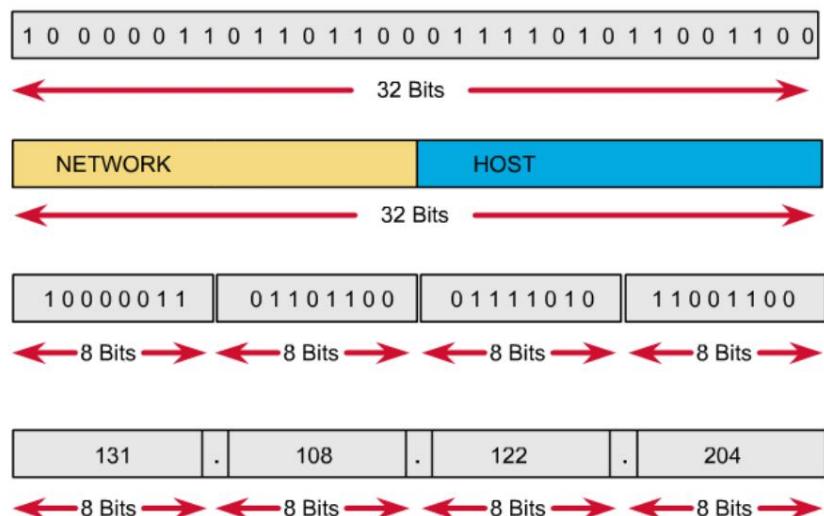


Network Topology



IP Address

IP address format

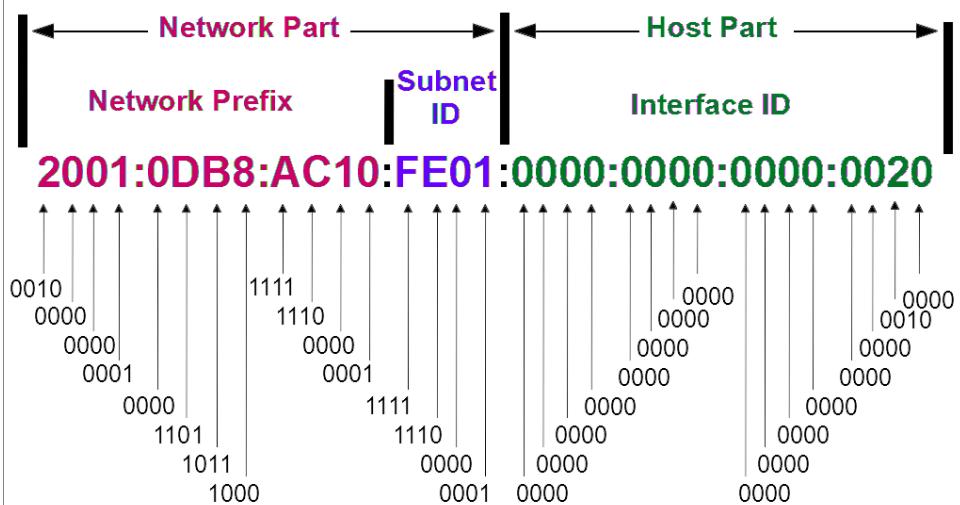


Class	Private Address Ranges
Class A	10.0.0.0 – 10.255.255.255
Class B	172.16.0.0 – 172.31.255.255
Class C	192.168.0.0 – 192.168.255.255
Loopback	127.0.0.0 – 127.255.255.255 (127.0.0.1)

IPv6 Address Structure

128 Bits, Expressed in Hex (Hexadecimal) with 3 parts

This is the usual breakdown but it can be broken down in other ways

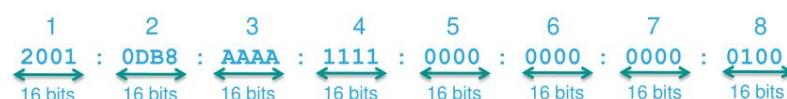


IPv6 Address Notation

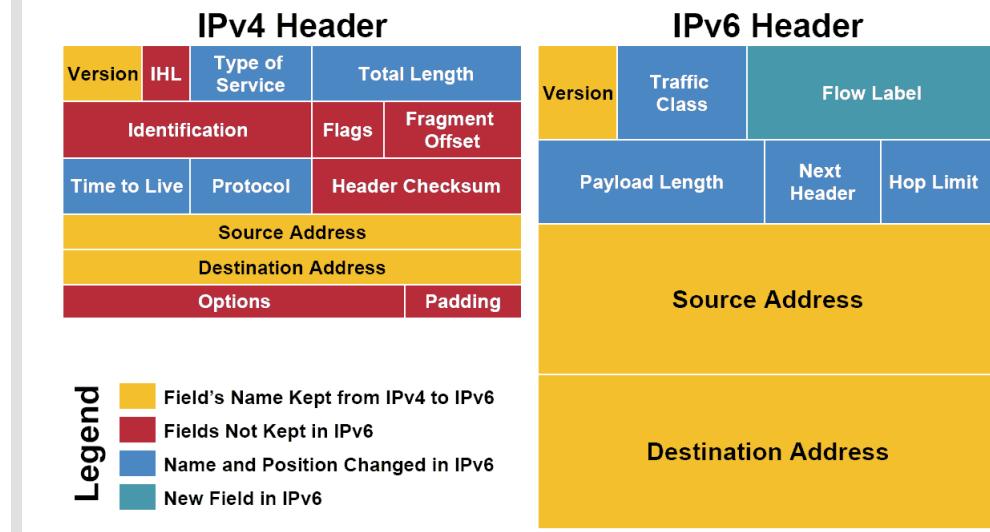
One Hex digit = 4 bits

Dec.	Hex.	Binary	Dec.	Hex.	Binary
0	0	0000	8	8	1000
1	1	0001	9	9	1001
2	2	0010	10	A	1010
3	3	0011	11	B	1011
4	4	0100	12	C	1100
5	5	0101	13	D	1101
6	6	0110	14	E	1110
7	7	0111	15	F	1111

2001:0DB8:AAAA:1111:0000:0000:0100:64



- IPv6 addresses are 128-bit addresses represented in:
 - Eight 16-bit segments or “hextets” (not a formal term)
 - Hexadecimal (non-case sensitive) between 0000 and FFFF



IP Range

IP Address Ranges

IP Address Class	First Octet Binary Value	First Octet Decimal Value	Possible Number of Hosts
Class A	1-126	<u>0</u> 0000001 to <u>0</u> 1111110*	16,777,214
Class B	128-191	<u>1</u> 0000000 to <u>1</u> 0111111	65,534
Class C	192-223	<u>1</u> 1000000 to <u>1</u> 1011111	254

*127 (01111111) is a Class A address reserved for loopback testing and cannot be assigned to a network.

Private IP ranges

- Often it is necessary to connect devices to the network, but not to the internet. RFC 1918 manages the private IP addresses that cannot appear on the internet, but are reserved for private use.

- Private IP ranges managed by IANA:

Class	From	To	No. Of hosts
1 x A class	10.0.0.0	10.255.255.255	$2^{24} = 16.777.216$
16 x B class	172.16.0.0	172.31.255.255	$2^{20} = 1.048.576$
256 x C class	192.168.0.0	192.168.255.255	$2^{16} = 65.536$

- example:

- 192.168.1.0/24 (mask: 255.255.255.0 | 256 hosts) - 256 networks
- 172.17.0.0/16 (mask: 255.255.0.0 | 65.536 hosts) 256 networks

What Do You Need To Know About Private IP Address?

Class

Private Address Ranges

Class A

10.0.0.0 – 10.255.255.255

Class B

172.16.0.0 – 172.31.255.255

Class C

192.168.0.0 – 192.168.255.255

Loopback

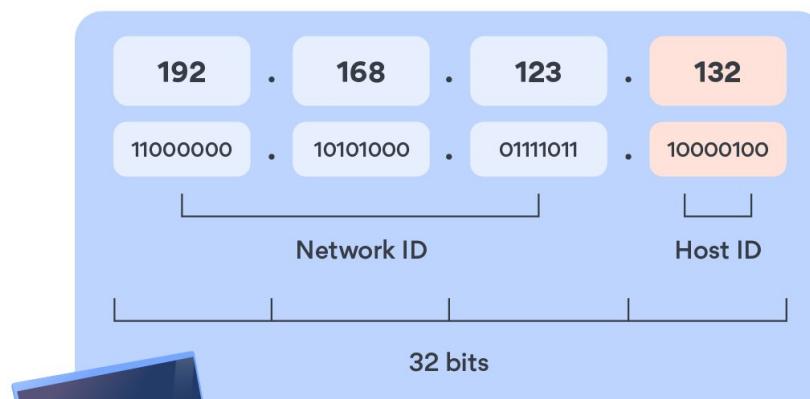
127.0.0.0 – 127.255.255.255
(127.0.0.1)

Subnet Mask

Subnet Mask

Prefix	Hosts	32-Borrowed=CIDR	$2^{\text{Borrowed}} = \text{Hosts}$	Binary=> dec = Prefix
.255	1	/32	0	11111111
.254	2	/31	1	11111110
.252	4	/30	2	11111100
.248	8	/29	3	11111000
.240	16	/28	4	11110000
.224	32	/27	5	11100000
.192	64	/26	6	11000000
.128	128	/25	7	10000000

IP address explained



The Default Subnet Masks (no subnets)

	1st octet	2nd octet	3rd octet	4th octet
Class A	Network	Host	Host	Host
Class B	Network	Network	Host	Host
Class C	Network	Network	Network	Host
Class A or /8	11111111	00000000	00000000	00000000
Class B or /16	11111111	11111111	00000000	00000000
Class C or /24	11111111	11111111	11111111	00000000

- A "1" bit in the subnet mask means that the corresponding bit in the IP address should be read as a network number
- A "0" bit in the subnet mask means that the corresponding bit in the IP address should be read as a host bit.
- /n "slash" tells us how many "1" bits are in the subnet mask.

The Subnet Mask

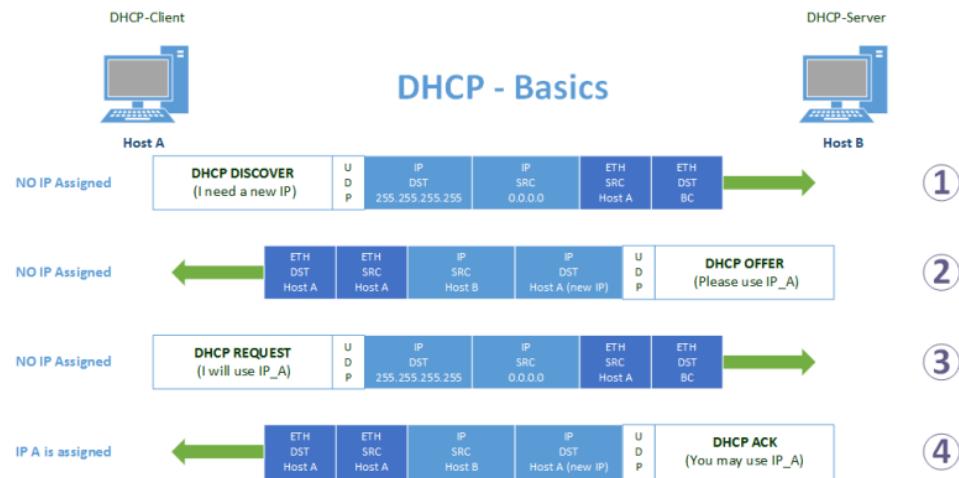
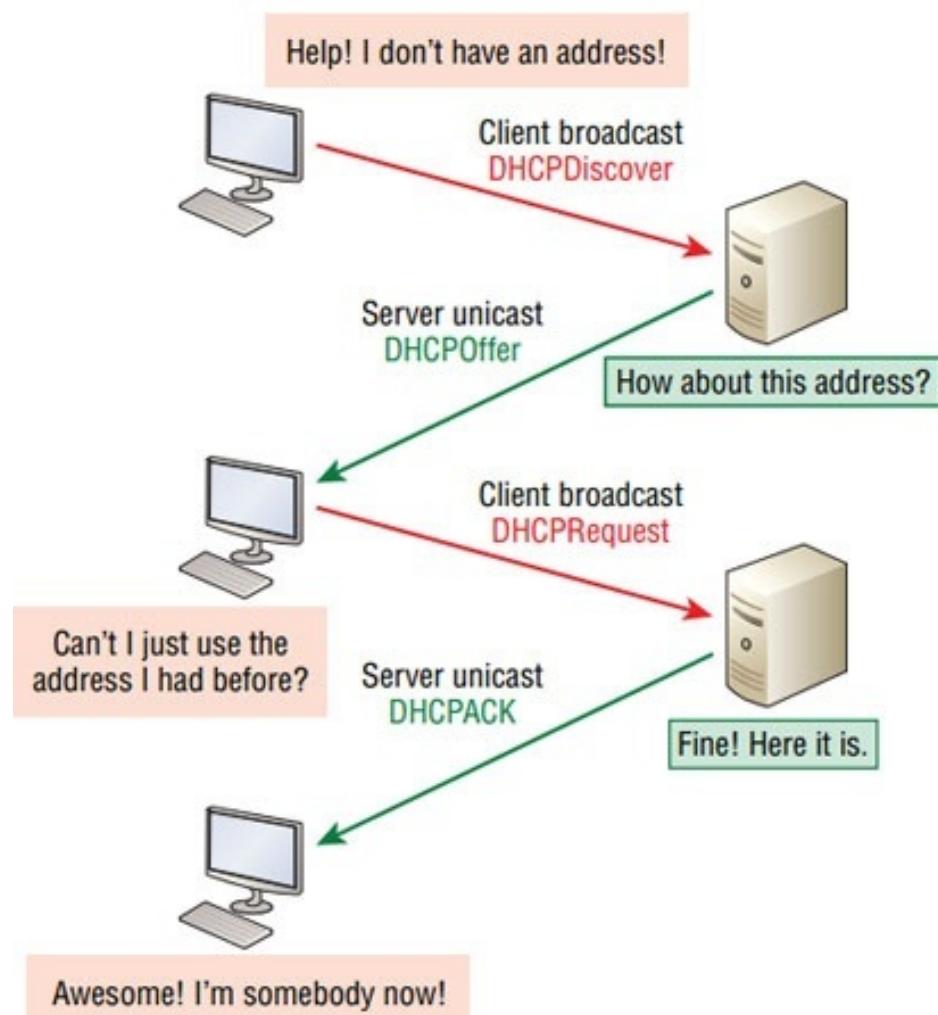
- Subnet Mask:
 - Let's not forget about the subnet mask.
 - Each class has a **default or "natural"** subnet mask based on the default number of bits used for the network and host portion.

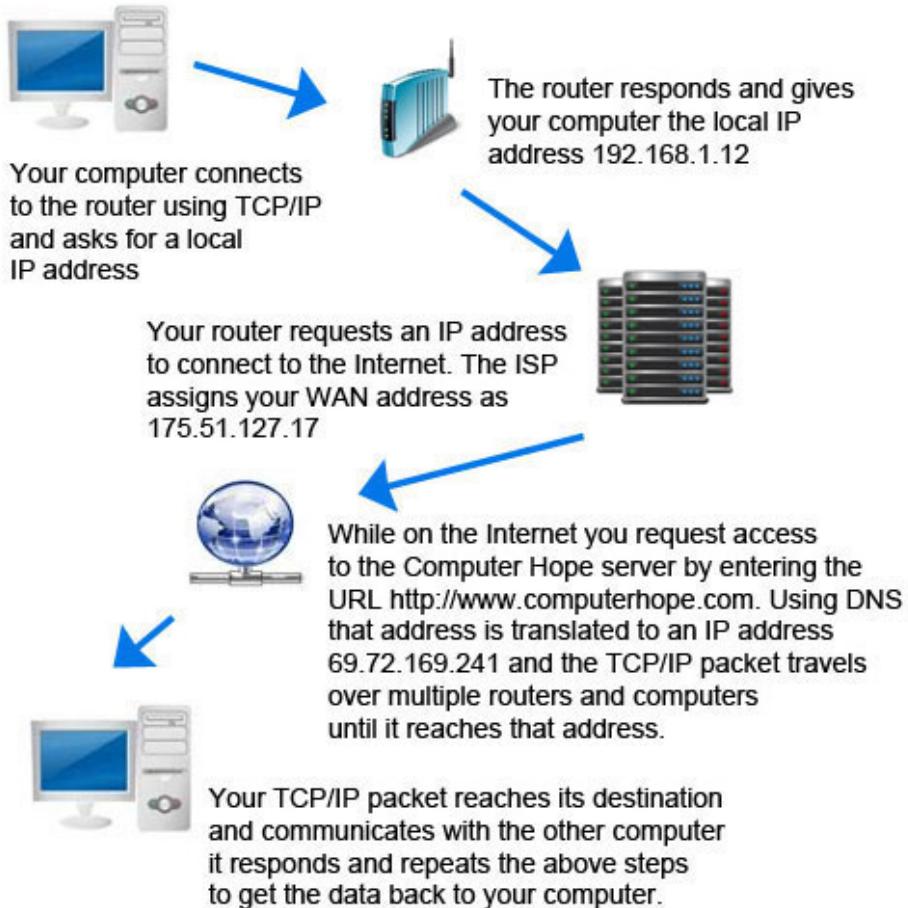
Class	Number of Network Bits	Number of Host Bits	Default Prefix	Default Subnet Mask
A	8	24	/8	255.0.0.0
B	16	16	/16	255.255.0.0
C	24	8	/24	255.255.255.0

IP Addressing Methods

Dynamic IP

DHCP





ComputerHope.com

APIPA

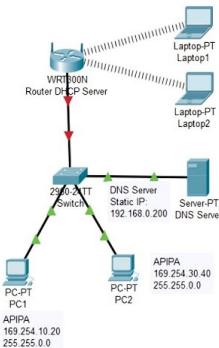
MAHA NETWORK
All about EDUCATION

APIPA Class B Private IP v4 Address

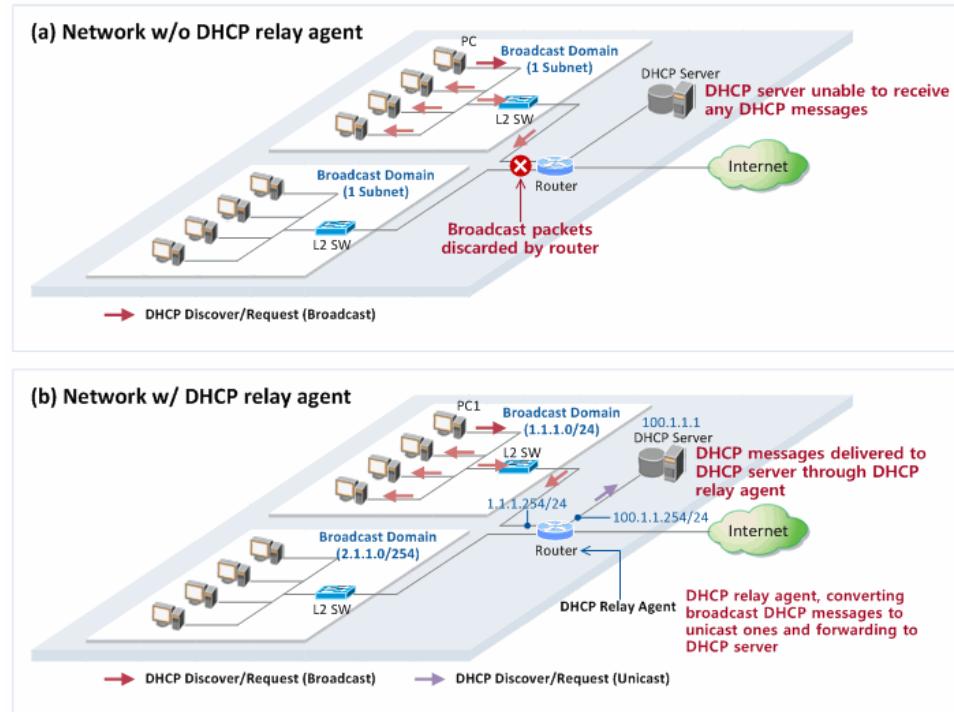
❖ Automatic Private IP Address

169.254. x. x

- When connection between **DHCP Server & N/W device (Switch)** goes **DOWN**, **APIPA** addresses are **AUTOMATICALLY** created on **END User Devices** like Desktops, PCs, Laptops, Printers etc.
- **END User Devices** who have **APIPA** addresses can **ONLY** communicate **INSIDE** the own **LOCAL N/W**
- **APIPA** addresses **DO NOT** go out of their **OWN N/W**
- **APIPA** addresses are **NOT ROUTABLE**
- If **APIPA** addresses are seen on **END Devices** than this is a **INTERNAL N/W** problem
- Check the **MEDIA or CABLE** between **DHCP server (Router) & N/W device (Switch)** inside **LOCAL N/W**



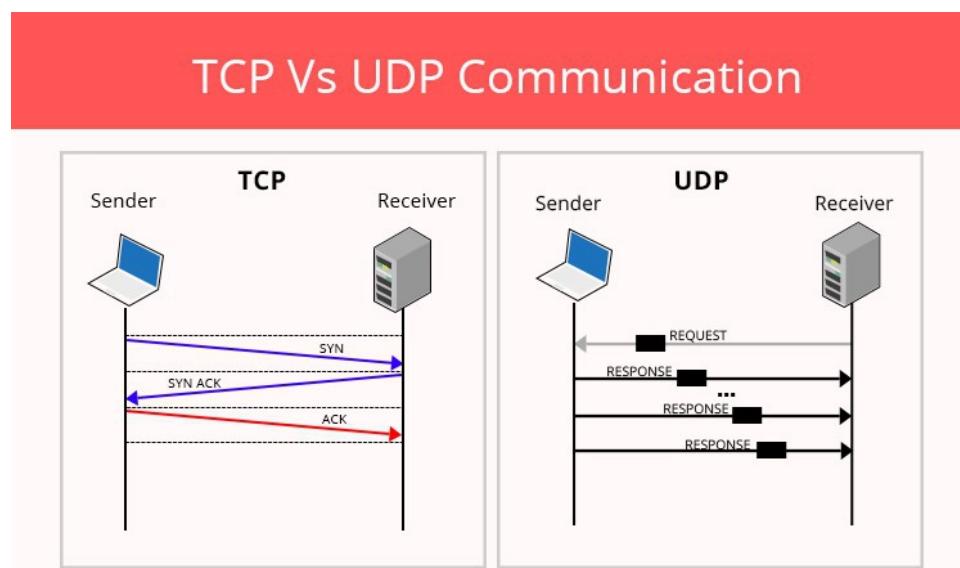
DHCP Relay = IP Helper

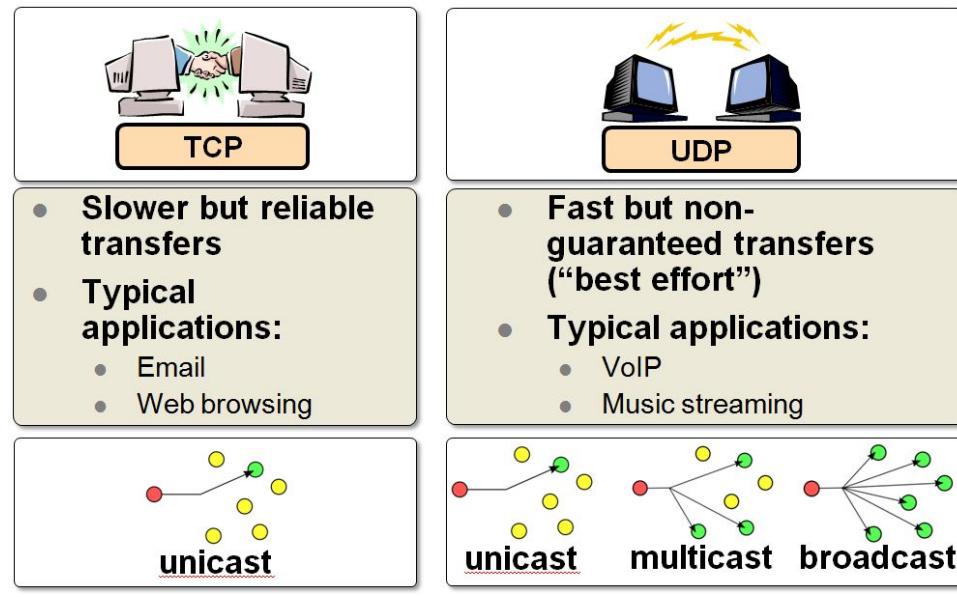


ARP(Address Resolution Protocol) / RARP

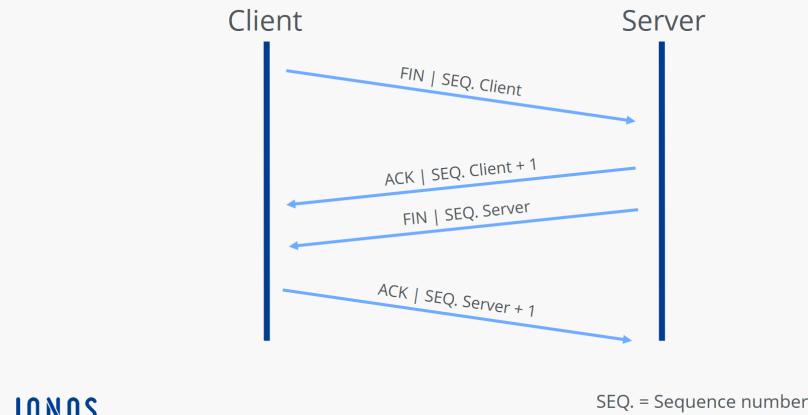
- ARP: resolve IP Address to MAC Address
- RARP: resolve MAC Address to IP Address

TCP vs UDP





TCP connection termination (TCP Teardown)

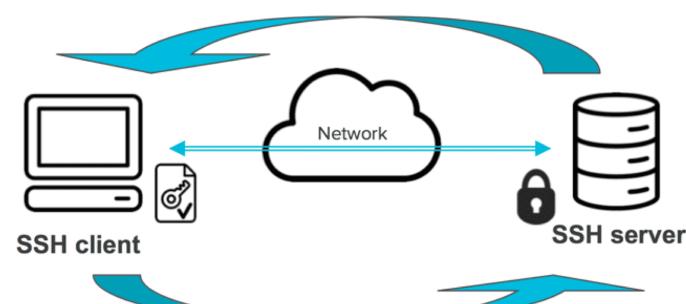


Ports

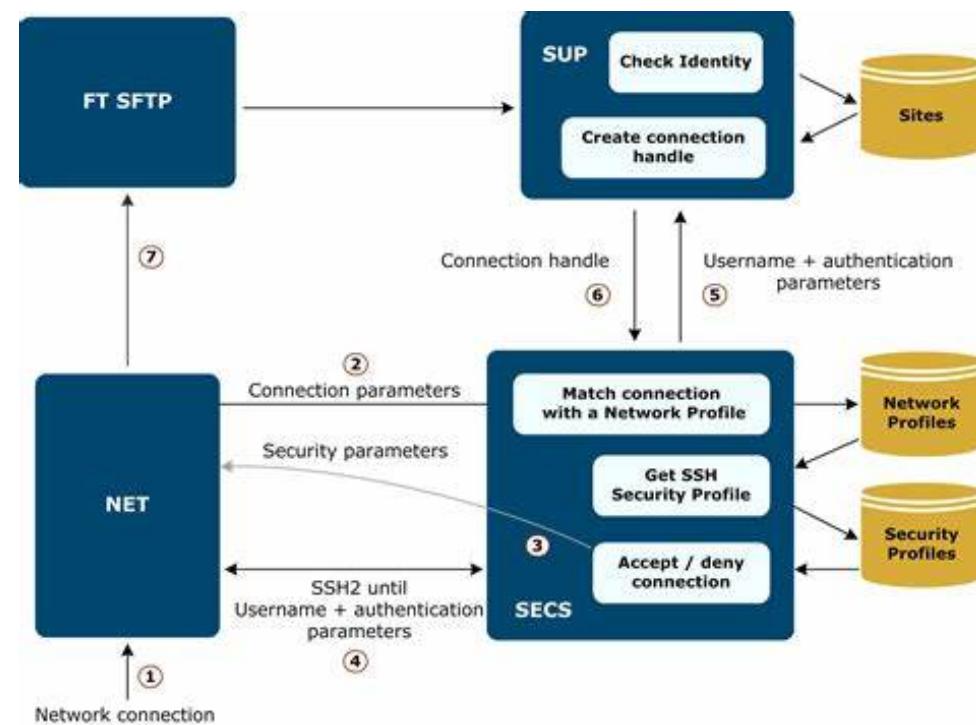
Port Number	Transport Protocol	Service Name	RFC
20, 21	TCP	File Transfer Protocol (FTP)	RFC 959
22	TCP and UDP	Secure Shell (SSH)	RFC 4250-4256
23	TCP	Telnet	RFC 854
25	TCP	Simple Mail Transfer Protocol (SMTP)	RFC 5321
53	TCP and UDP	Domain Name Server (DNS)	RFC 1034-1035
67, 68	UDP	Dynamic Host Configuration Protocol (DHCP)	RFC 2131
69	UDP	Trivial File Transfer Protocol (TFTP)	RFC 1350
80	TCP	HyperText Transfer Protocol (HTTP)	RFC 2616
110	TCP	Post Office Protocol (POP3)	RFC 1939
119	TCP	Network News Transport Protocol (NNTP)	RFC 8977
123	UDP	Network Time Protocol (NTP)	RFC 5905
135-139	TCP and UDP	NetBIOS	RFC 1001-1002
143	TCP and UDP	Internet Message Access Protocol (IMAP4)	RFC 3501
161, 162	TCP and UDP	Simple Network Management Protocol (SNMP)	RFC 1901-1908, 3411-3418
179	TCP	Border Gateway Protocol (BGP)	RFC 4271
389	TCP and UDP	Lightweight Directory Access Protocol	RFC 4510
443	TCP and UDP	HTTP with Secure Sockets Layer (SSL)	RFC 2818
500	UDP	Internet Security Association and Key Management Protocol (ISAKMP) / Internet Key Exchange (IKE)	RFC 2408 - 2409
636	TCP and UDP	Lightweight Directory Access Protocol over TLS/SSL (LDAPS)	RFC 4513
989/990	TCP	FTP over TLS/SSL	RFC 4217

SSH(Secure Shell)

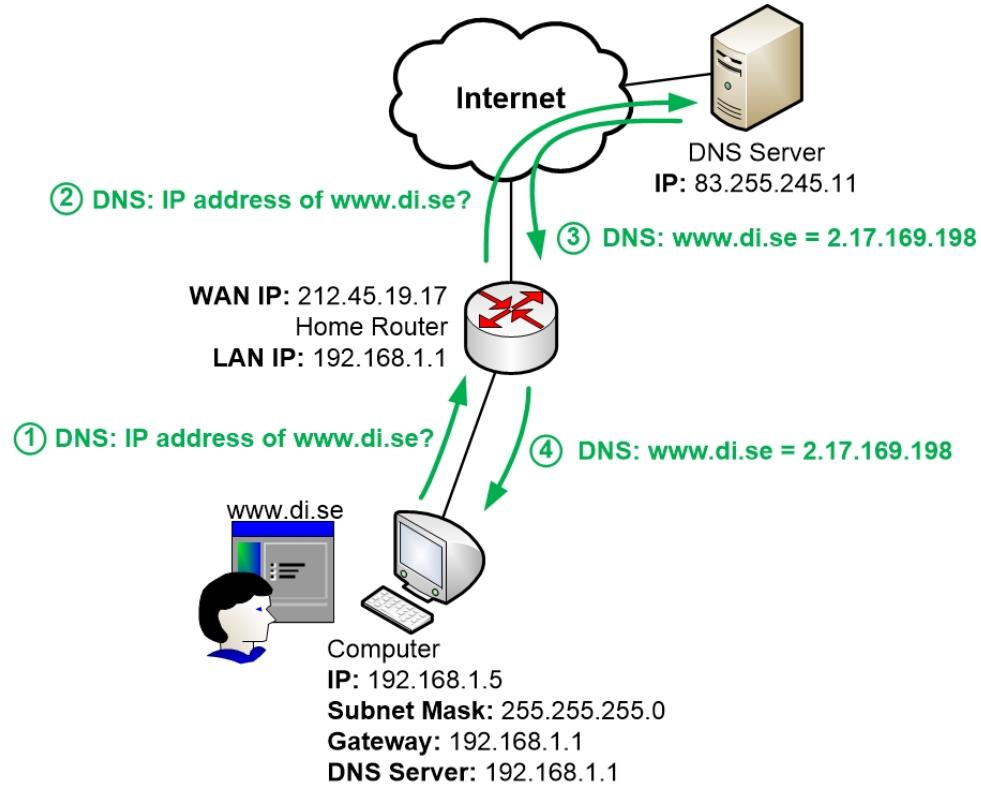
1) **Server authentication:**
Server proves its identity to the client



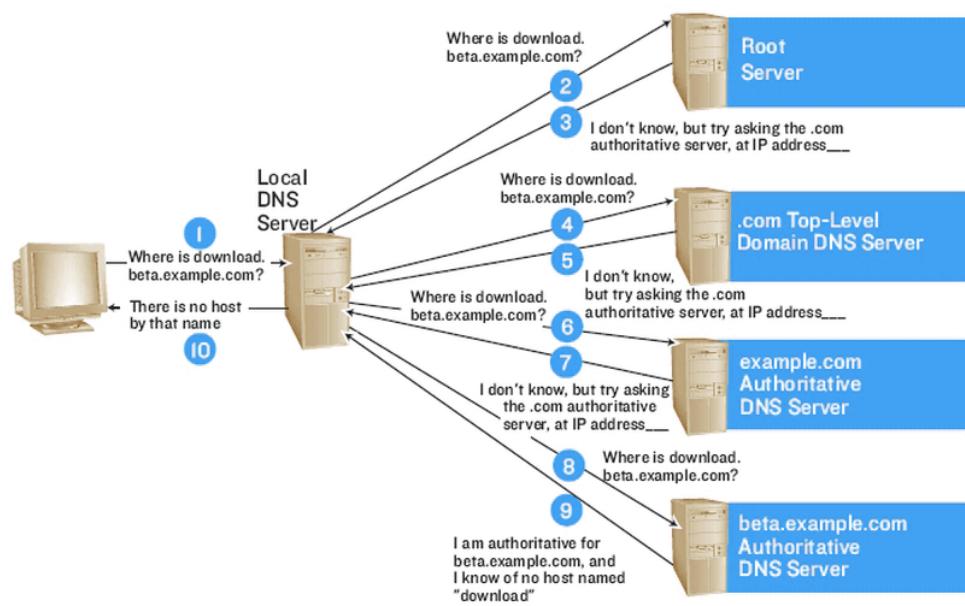
2) **User authentication:**
Client proves user's identity to the server

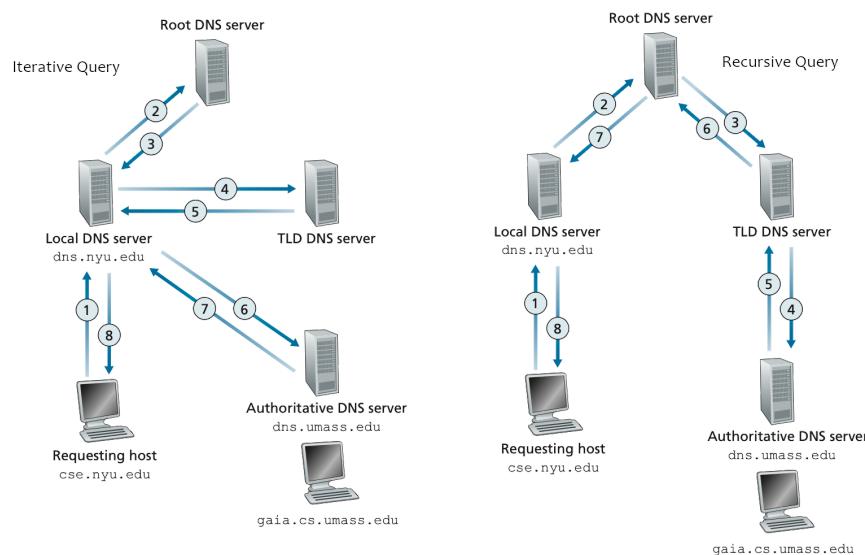
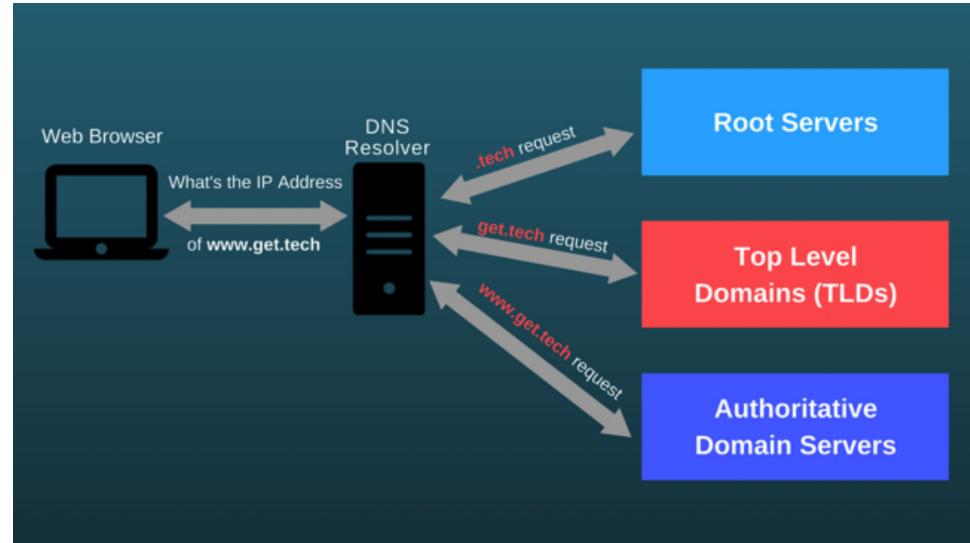


DNS(Domain Name System)



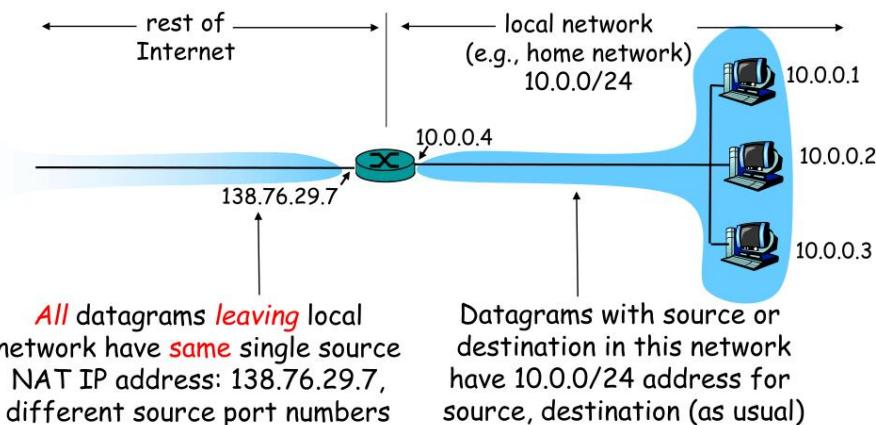
HOW DNS WORKS

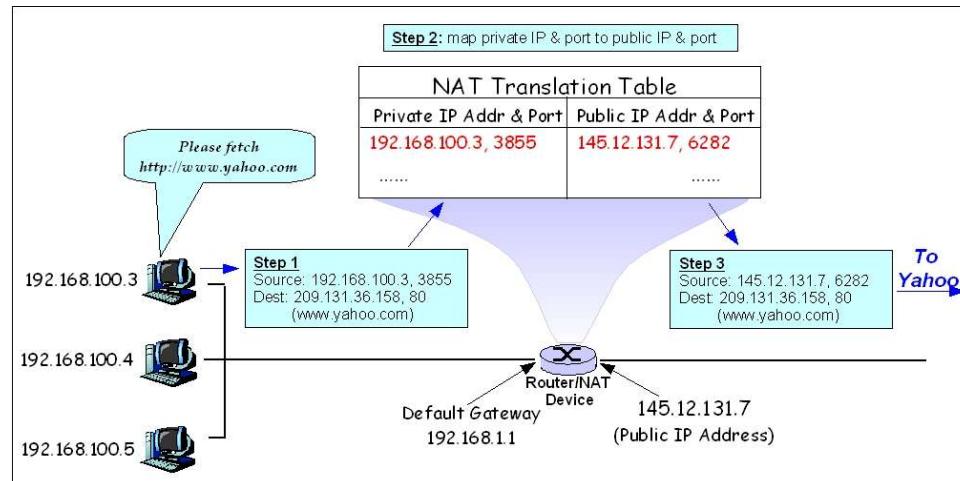




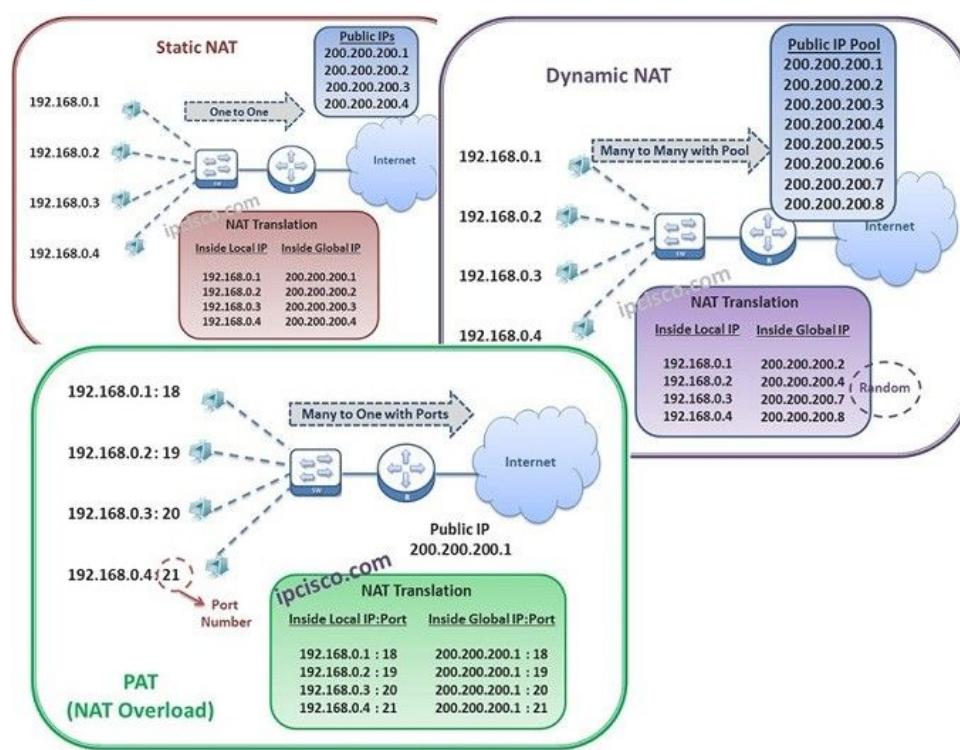
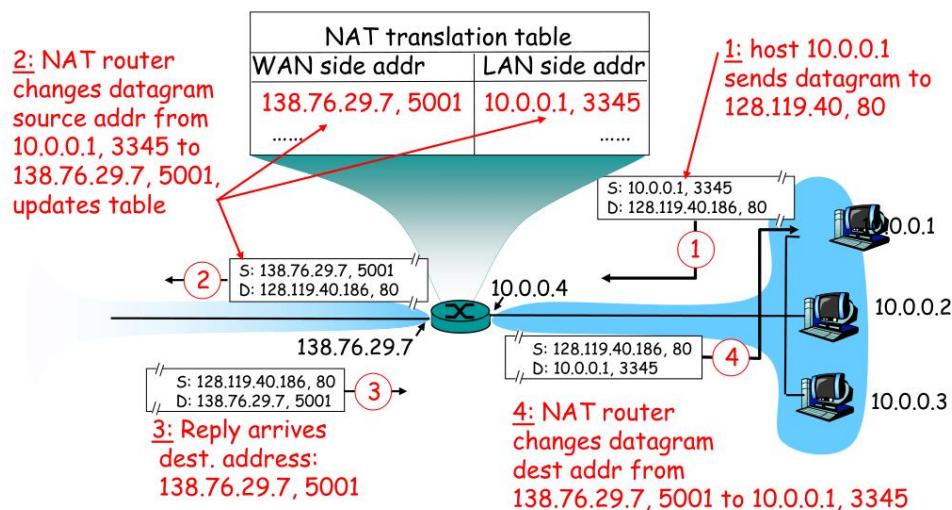
NAT(Network Address Translation)

NAT: Network Address Translation

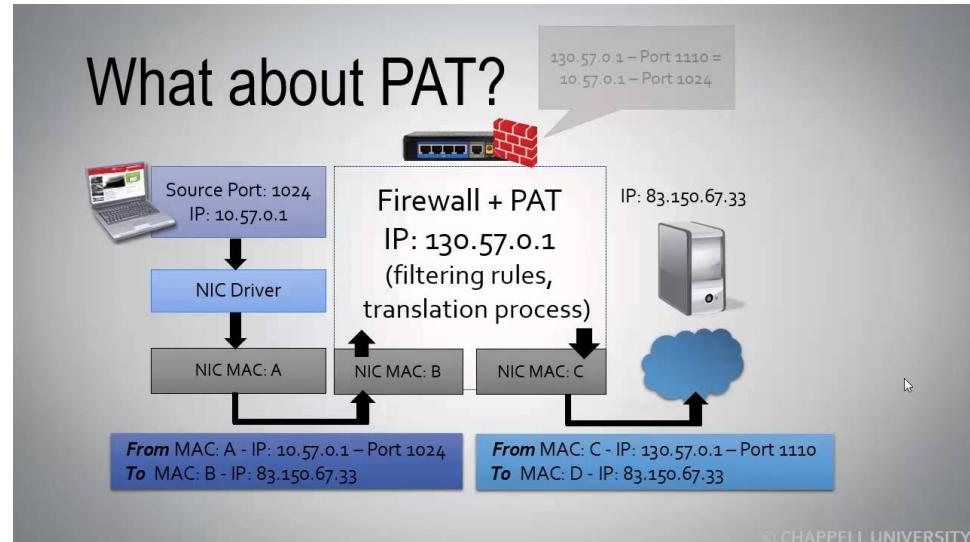




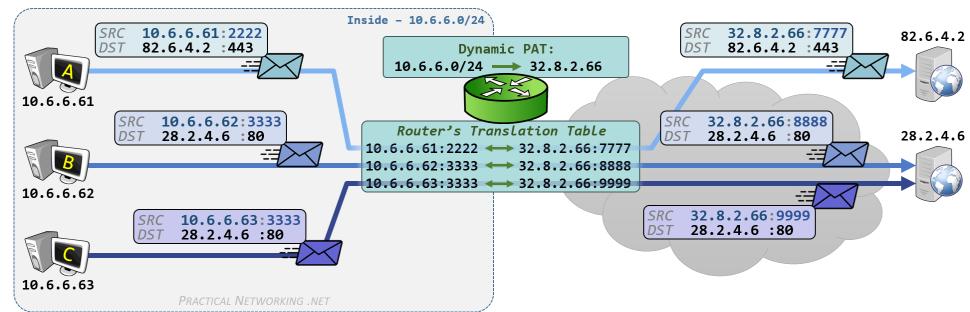
NAT: Network Address Translation



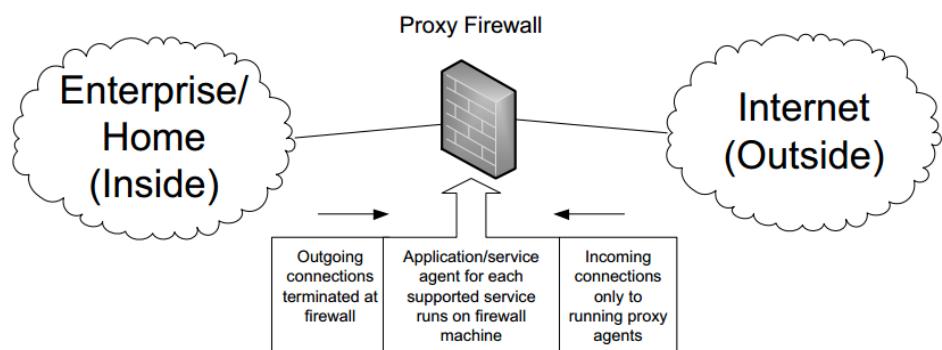
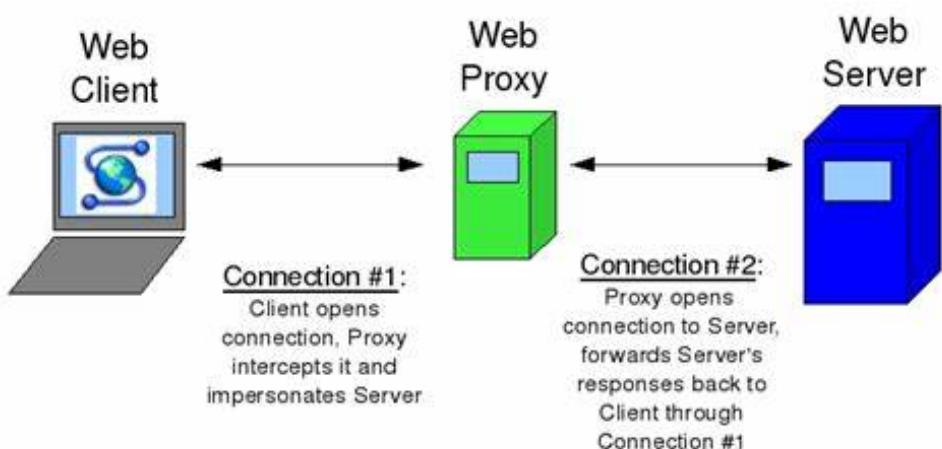
PAT (Port Address Translation)



© CHAPPELL UNIVERSITY



Proxy



Proxy Cache

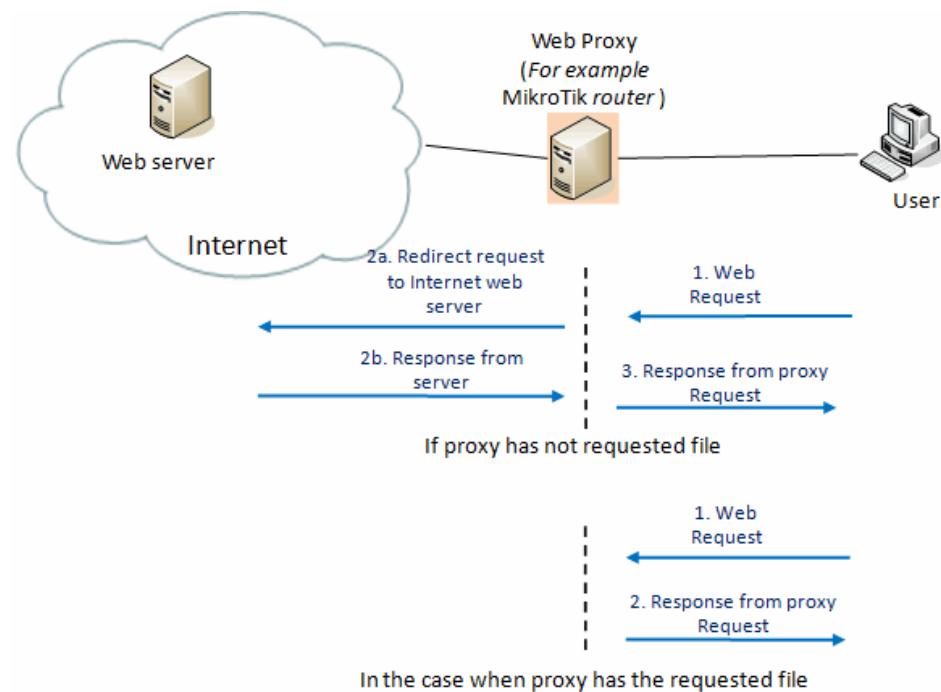
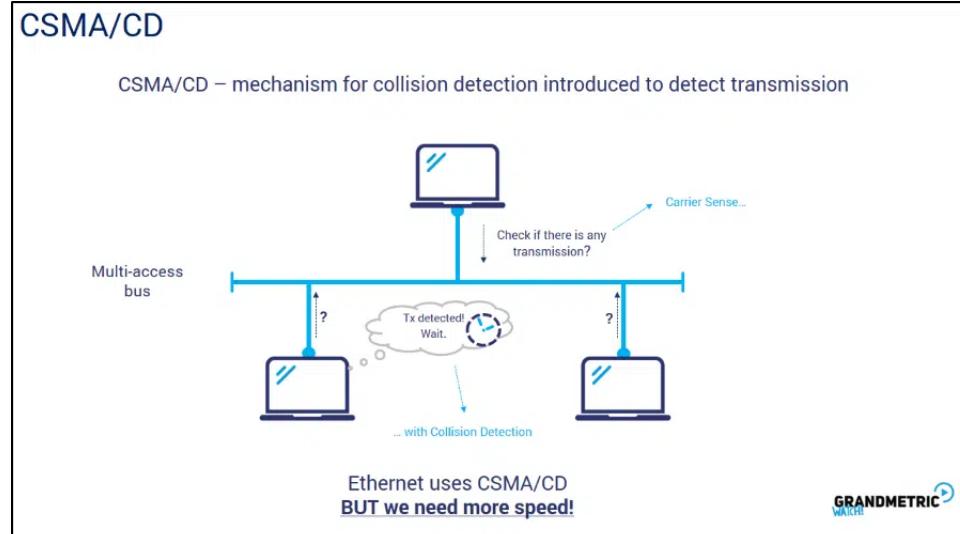
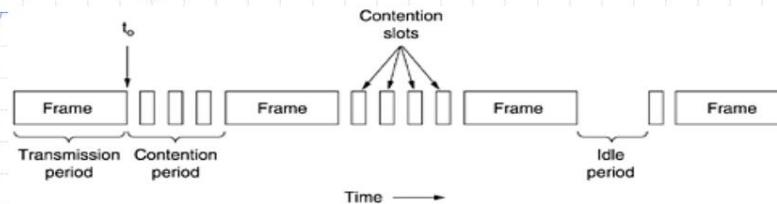


Figure 10.1. Web proxy basic operation scheme

CSMA/CD

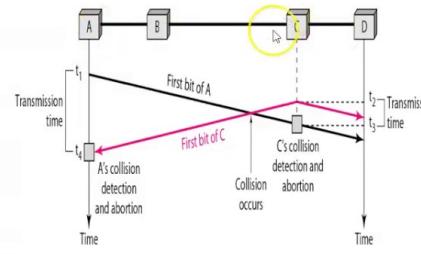


CSMA/CD

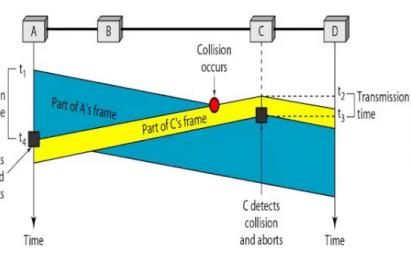


- ◆ Sense the channel
- ◆ Stop sending when detecting collision
- ◆ After collision wait a random amount of time and try again.

Collision in CSMA/CD



Collision of the first bit in CSMA/CD



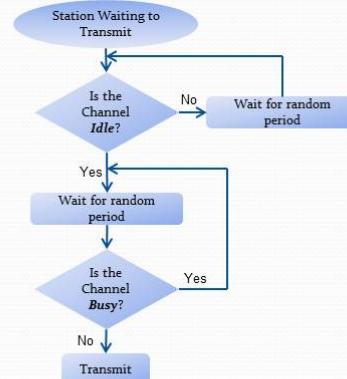
Collision and abortion in CSMA/CD

RECORDED WITH SCREENCASTOMATIC

CSMA/CA

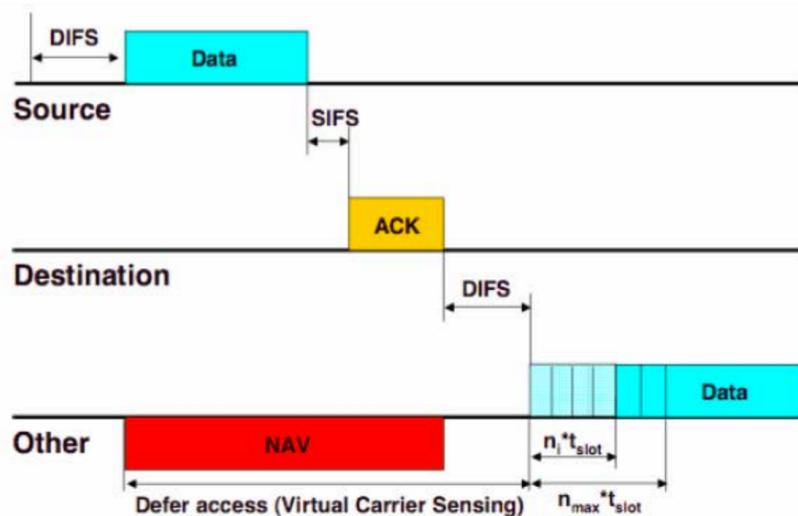
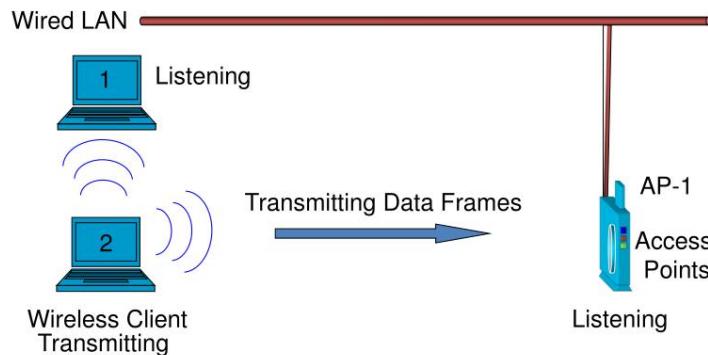
CSMA/CA

- CSMA/CA is a wireless network multiple access method in which:
 - A carrier sensing scheme is used.
 - A node wishing to transmit data has to first listen to the channel for a predetermined amount of time whether or not another node is transmitting on channel within the wireless range. If the channel is sensed as “idle”, then the node is permitted to begin the transmission process. If the channel is sensed as “busy”, the node defers its transmission for a random period of time.
 - State of channel “Idle” or “Busy” is based on CS mechanism, which will be explained later in the presentation

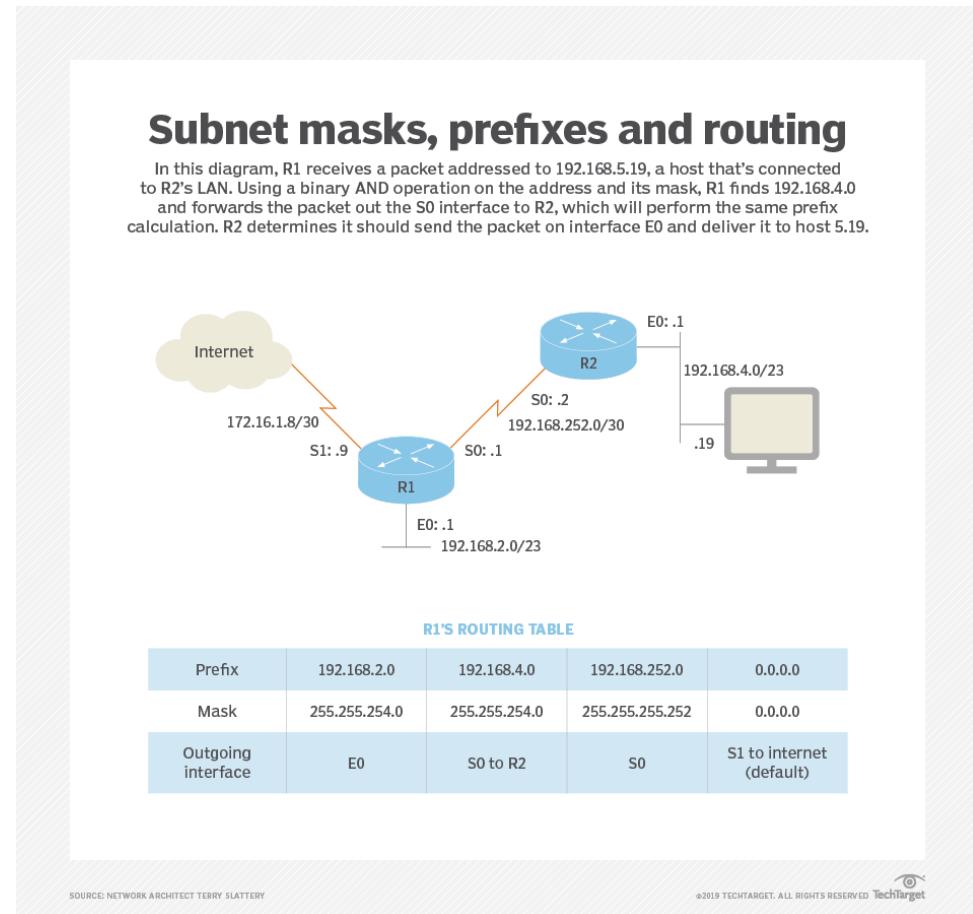


CSMA/CA Collision Handling

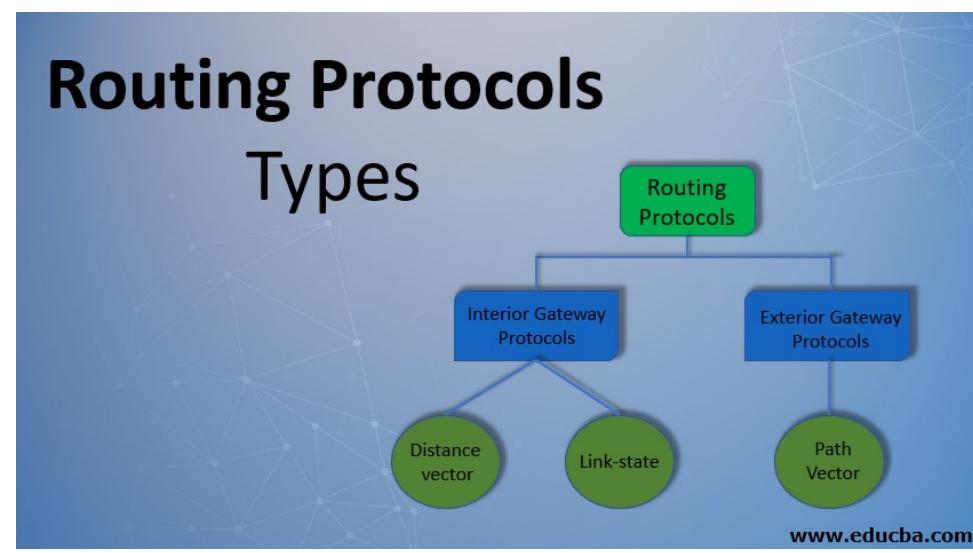
- 802.11 standard employs half-duplex radios-radios capable of transmission or reception-but not both simultaneously

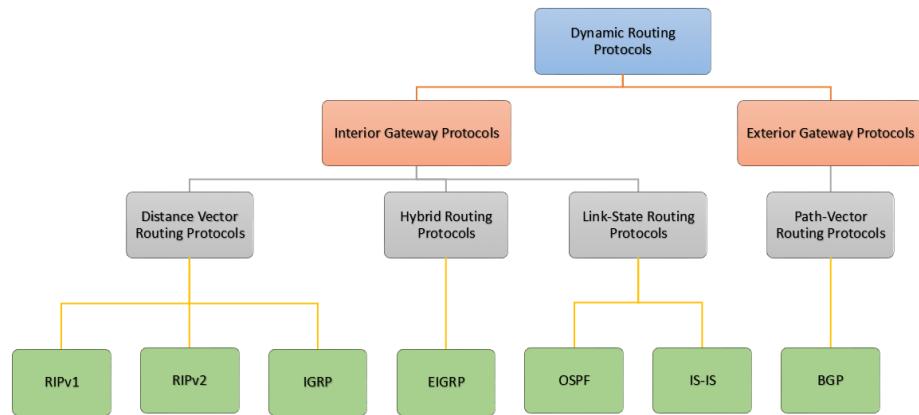


Routing Table



Routing Protocol





Routing Protocols for IP Networks

Protocol	Type	Scalability	Metric	IP classes
RIP-1	Distance vector	Small	Hop count	Classful
RIP-2	Distance vector	Small	Hop count	Classless
OSPF-2	Link state	Large	Cost	Classless
IS-IS	Link state	Very large	Cost	Classless
IGRP	Distance vector	Medium	Bandwidth, delay, load, MTU, reliability	Classful
EIGRP	Dual	Large	Bandwidth, delay, load, MTU, reliability	Classless
BGP	Distance vector	Large	Vector of attributes	Classless



7

Package

Components of a packet

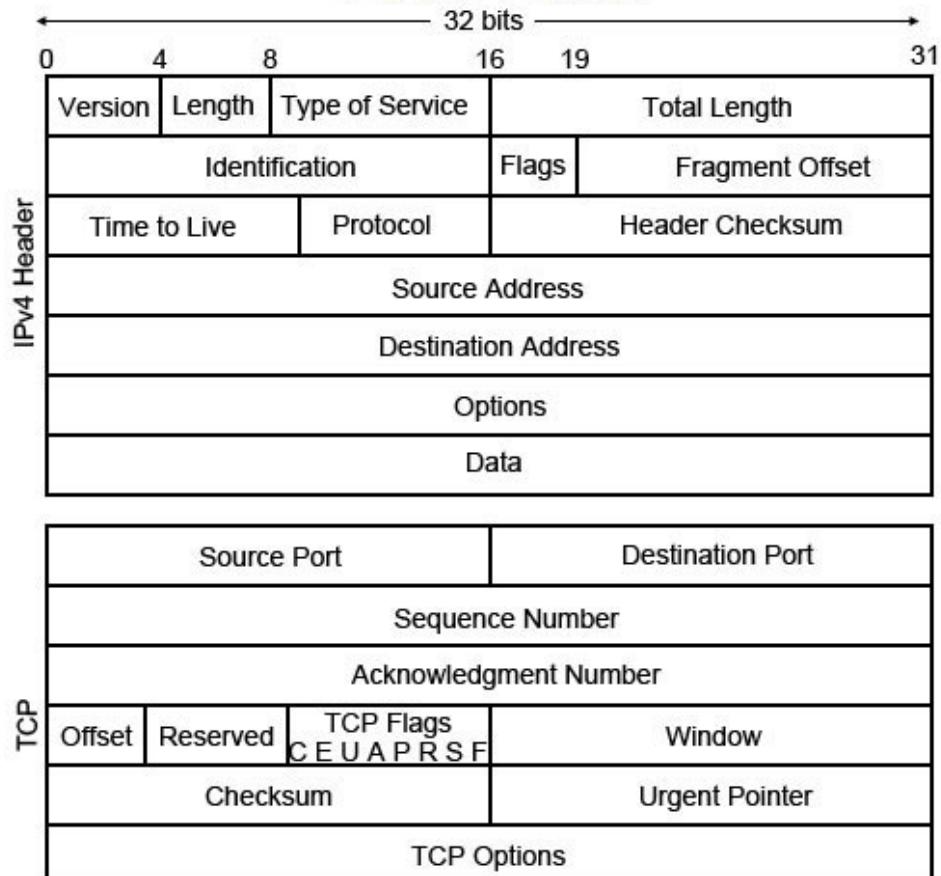
In its simplest form you can think of a packet consisting of three parts

Source Address	Destination Address	Data
----------------	---------------------	------

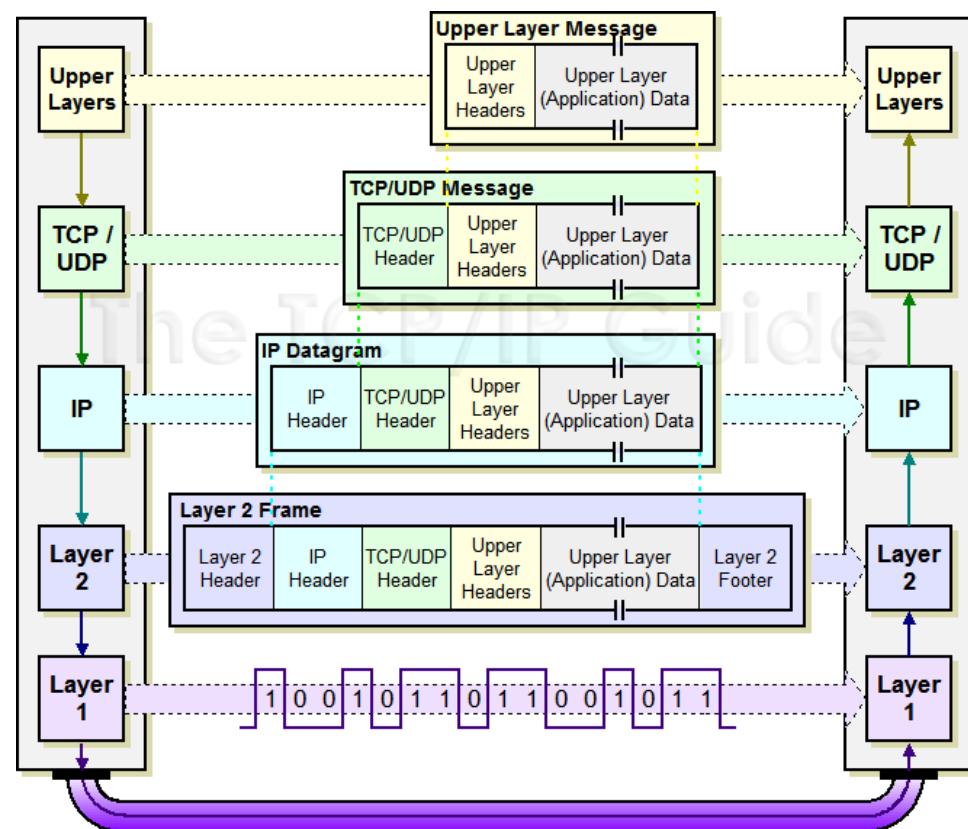
In principle there are slightly more components that form part of these sections

Source Address	Destination Address	Data
Source IP Address	Destination IP Address	Data
Source Mac Address	Destination Mac Address	Packet Sequence Number
Source Port Number	Destination Port Number	

TCP/IP Packet

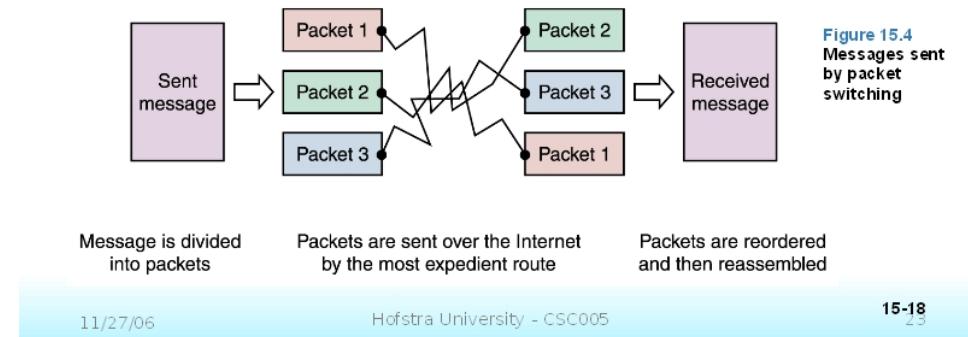


ComputerHope.com

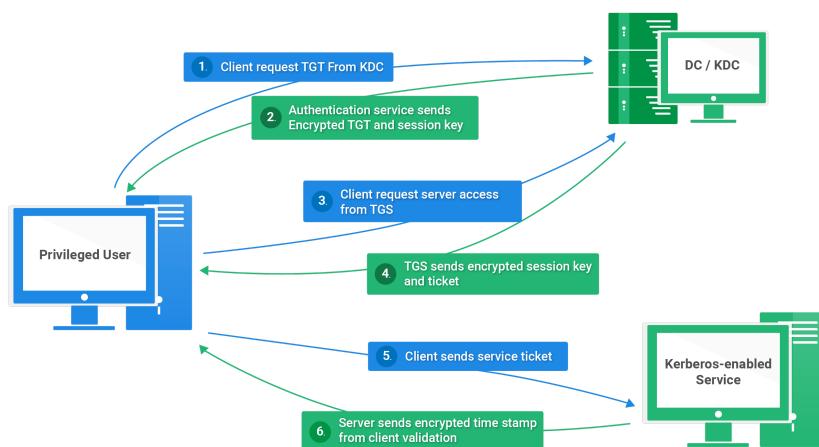
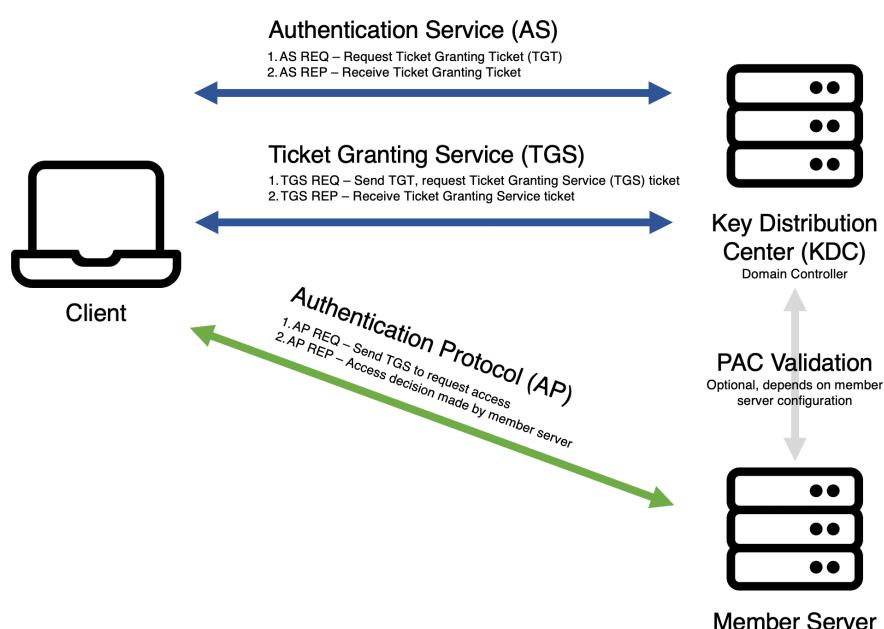


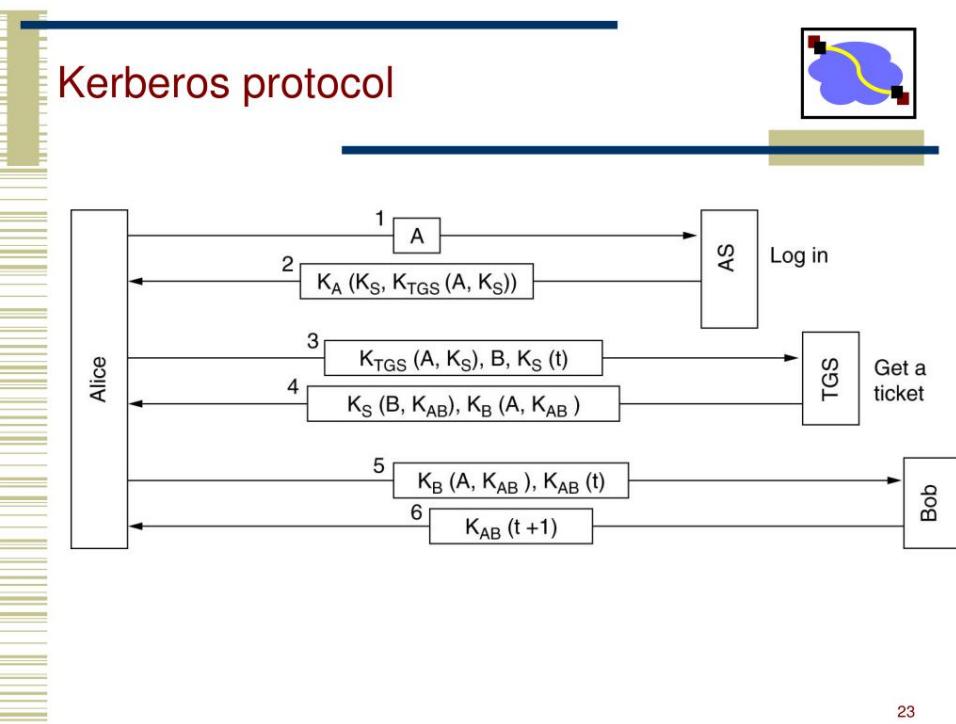
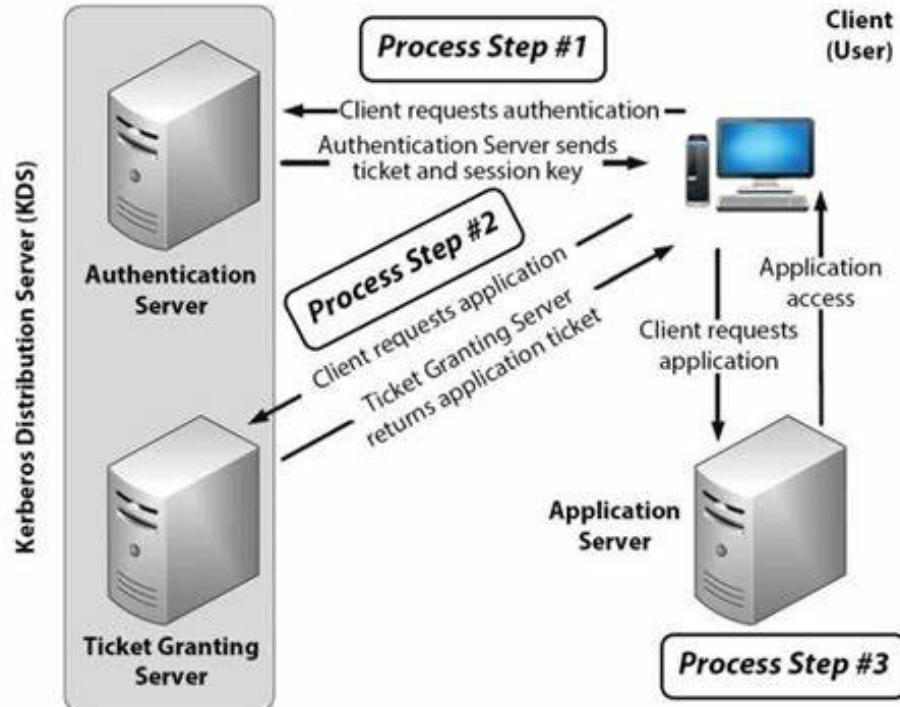
Packet Switching

- To improve the efficiency of transferring information over a shared communication line, messages are divided into fixed-sized, numbered **packets**
- Network devices called routers are used to direct packets between networks



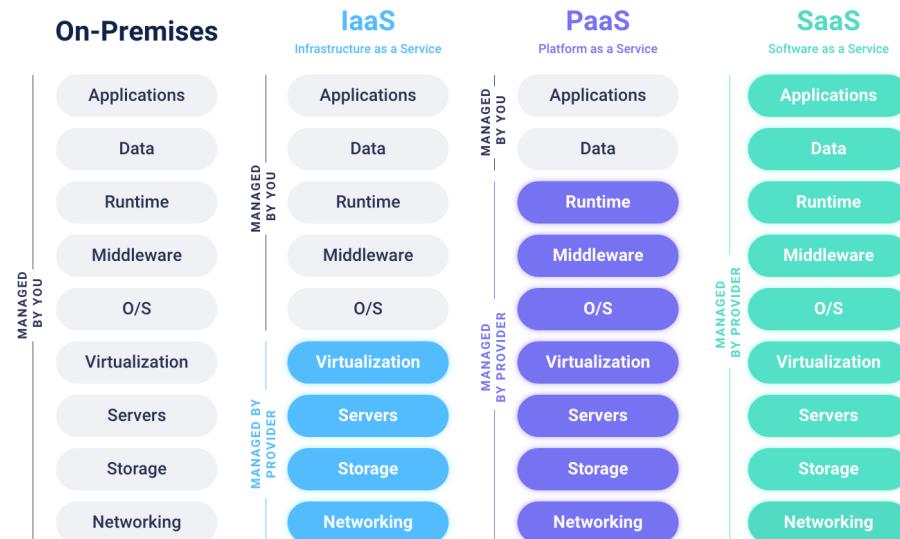
Kerberos

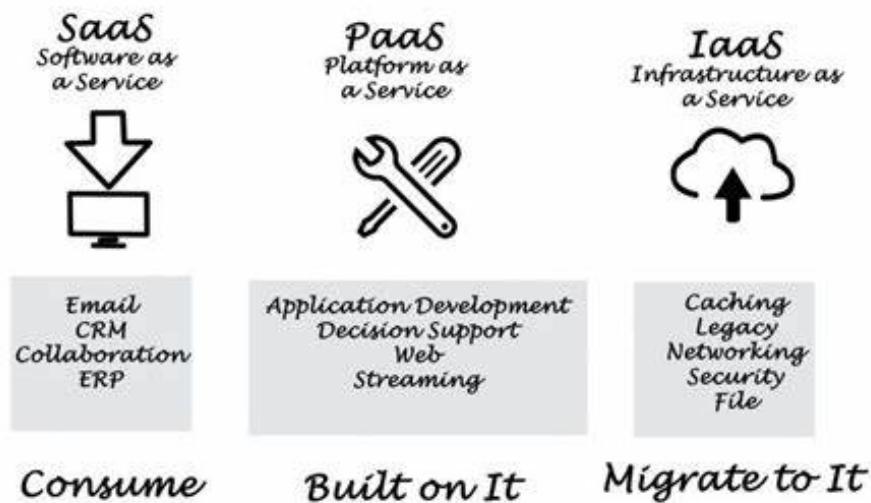


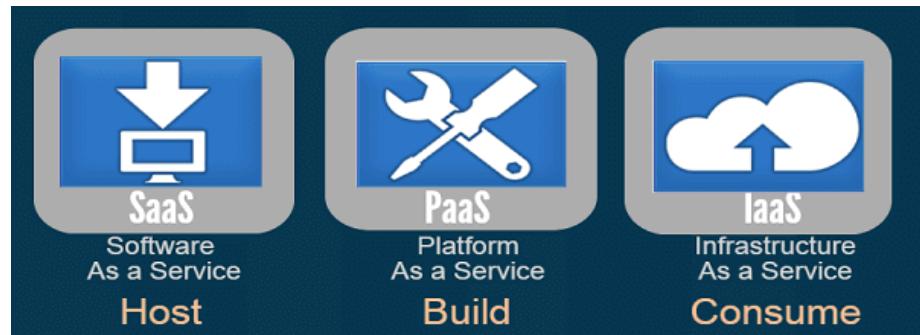


23

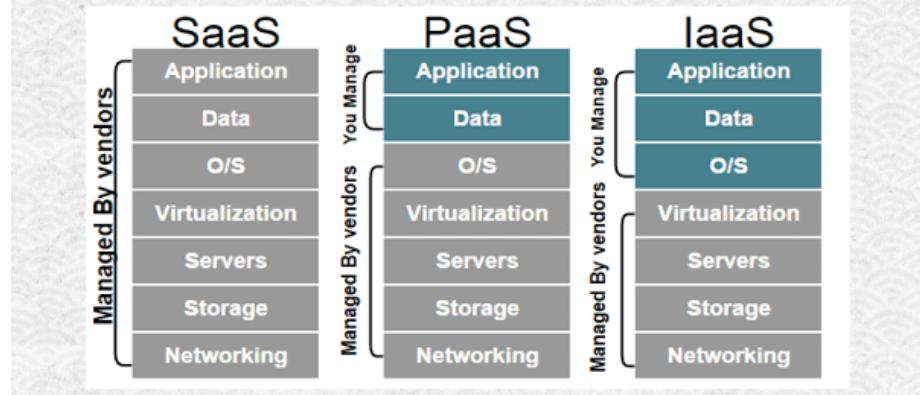
IaaS Paas Saas



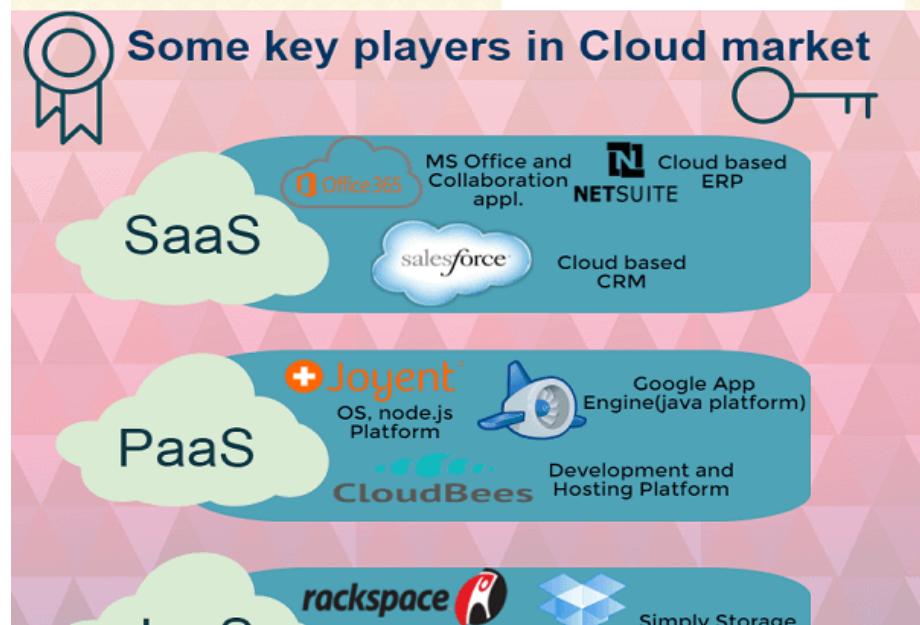
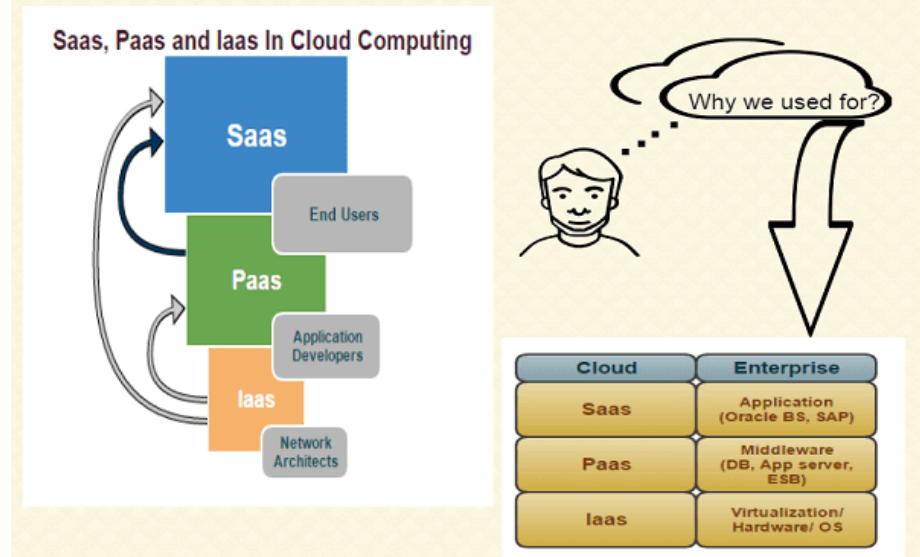




Difference between SaaS, PaaS and IaaS

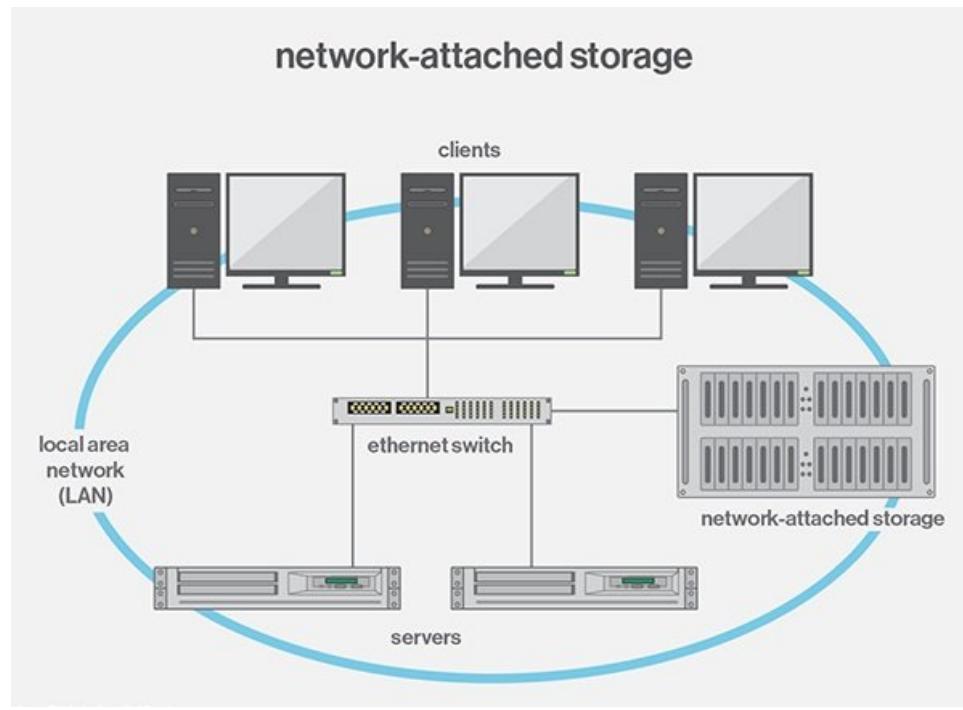


How Structured in Cloud Computing?

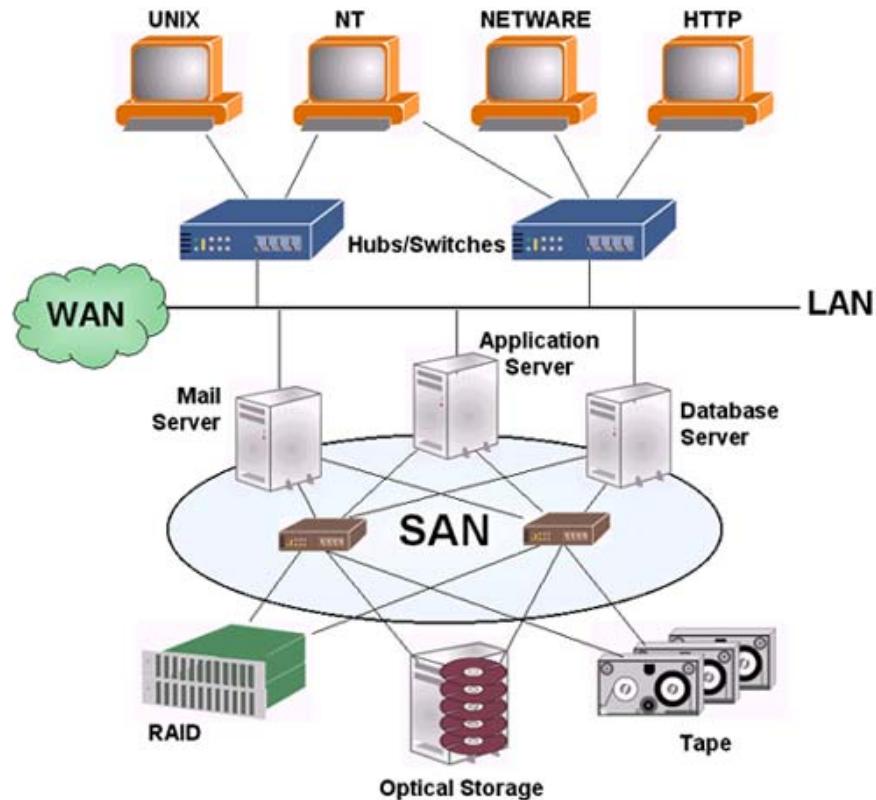




NAS vs SAN



① Storage Area Networks



Source: allSAN Report 2001

Copyright © 2000 allSAN.com Inc



SAN components

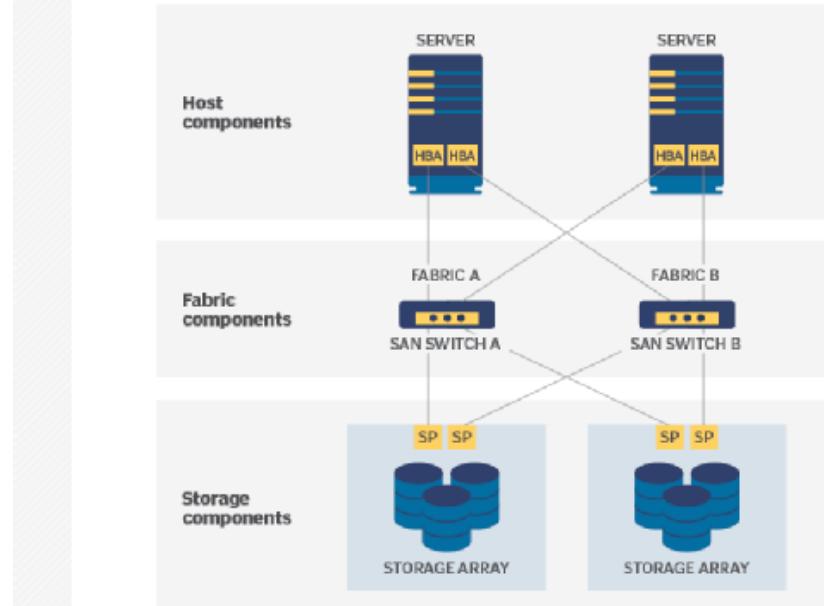


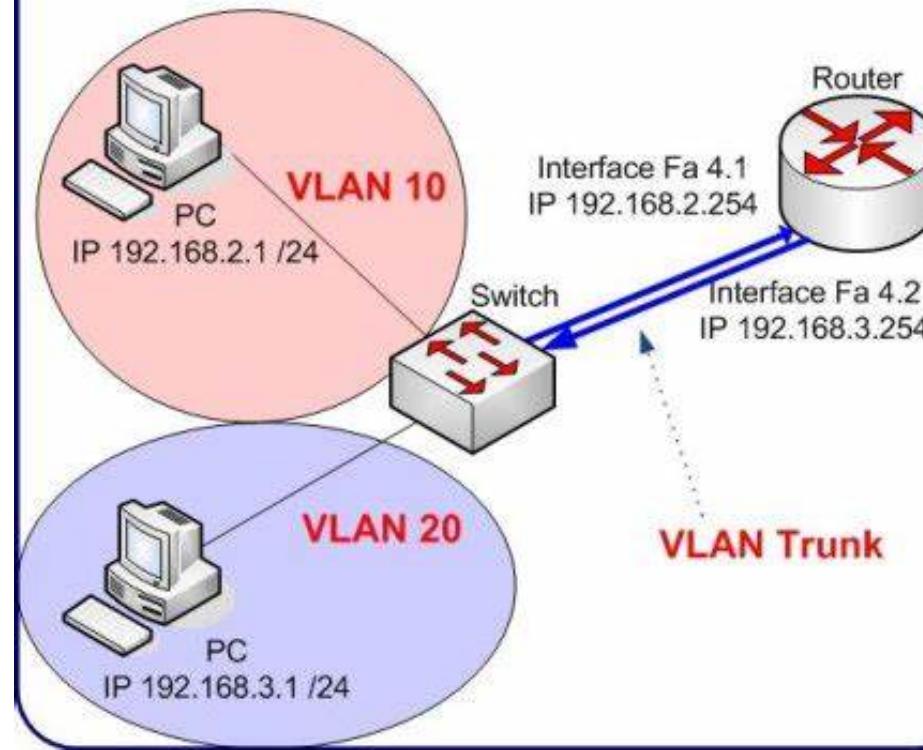
ILLUSTRATION: MAGLARA/ANDREW STOCK

©2000 TECHTARGET. ALL RIGHTS RESERVED



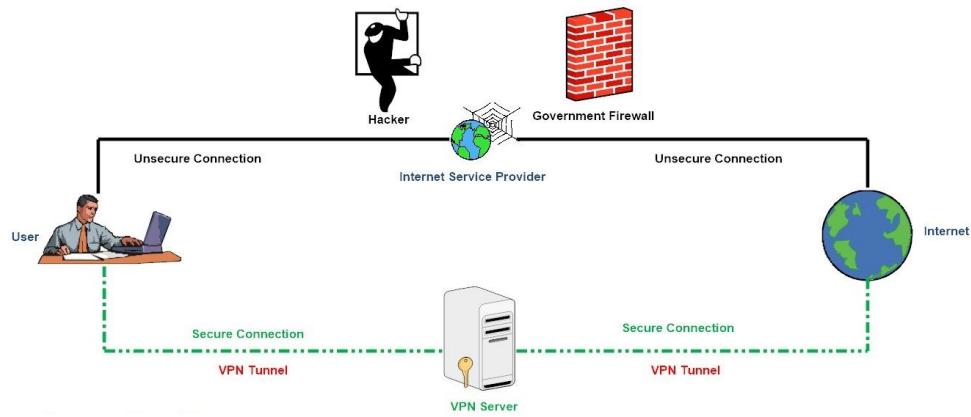
VLAN

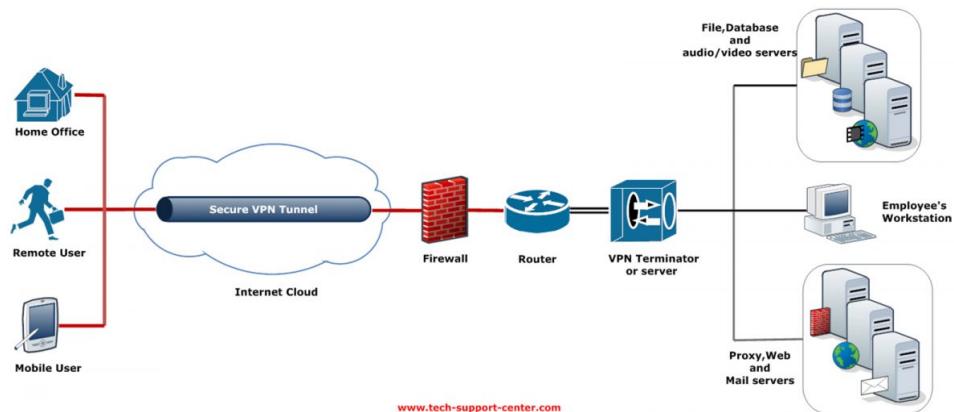
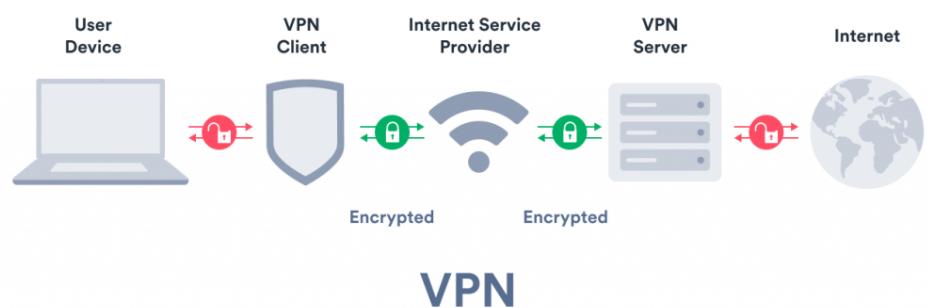
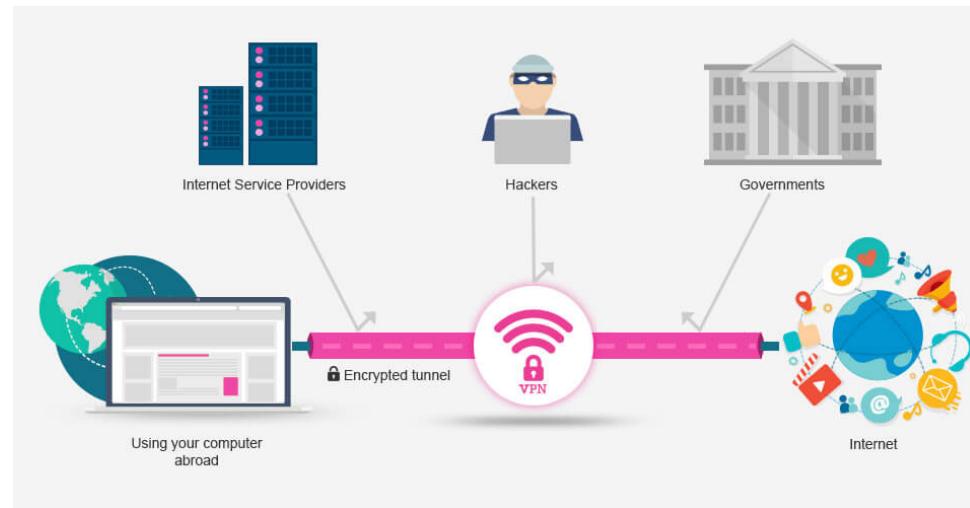
Sample network using VLANs



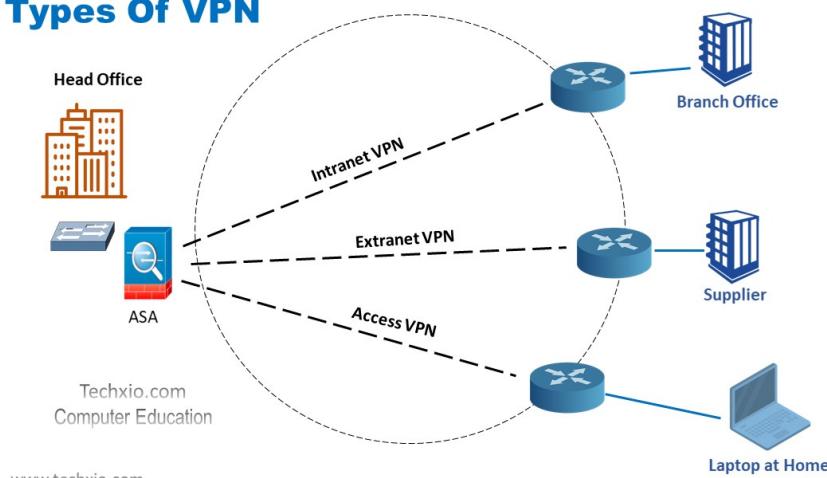
VPN

How VPN Works

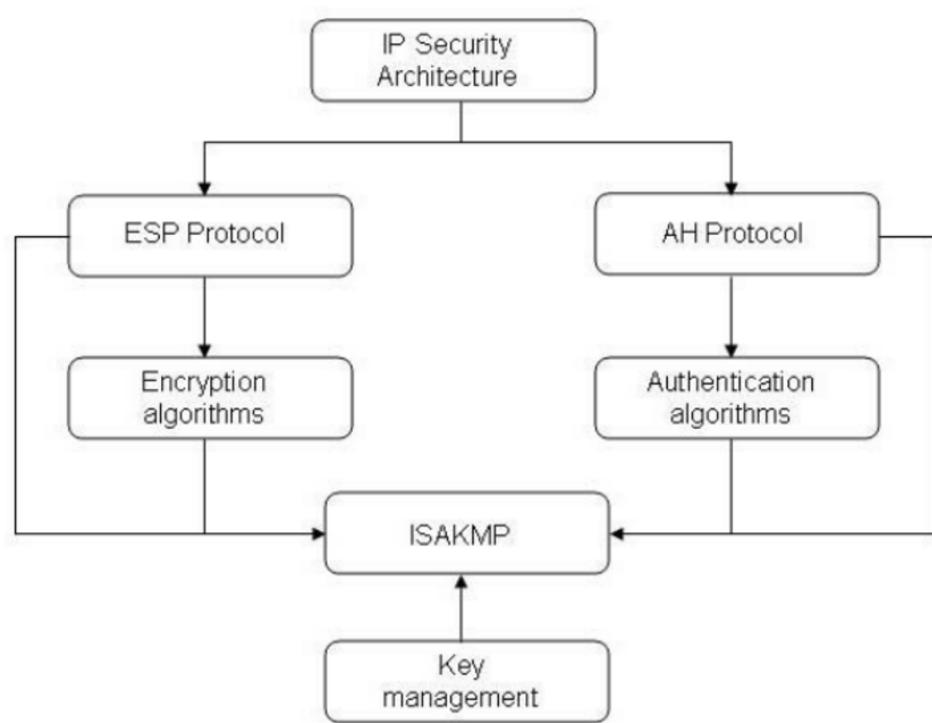




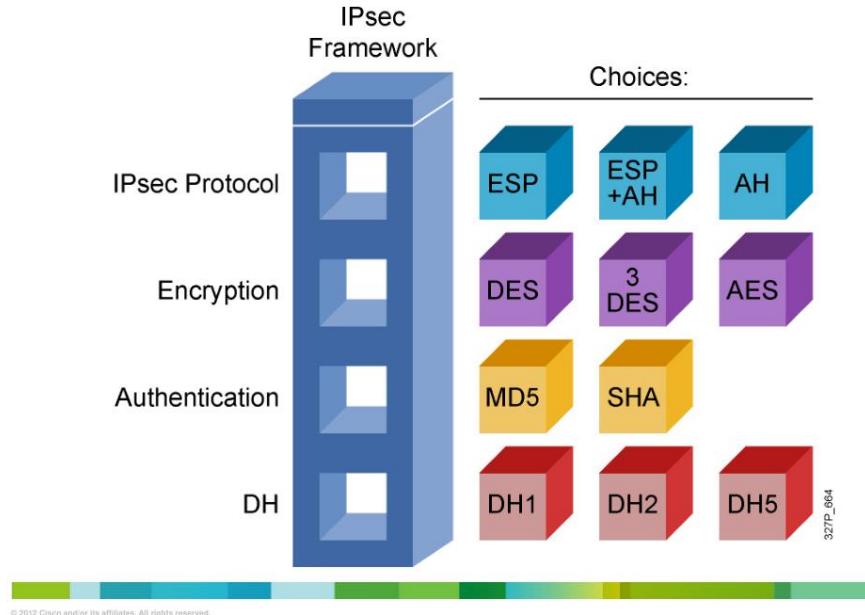
Types Of VPN

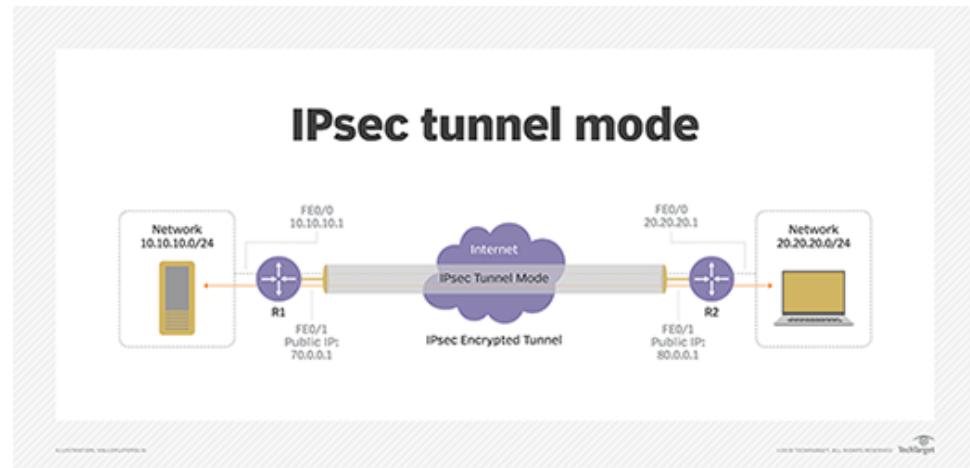


IPSec



IPsec Protocol Framework





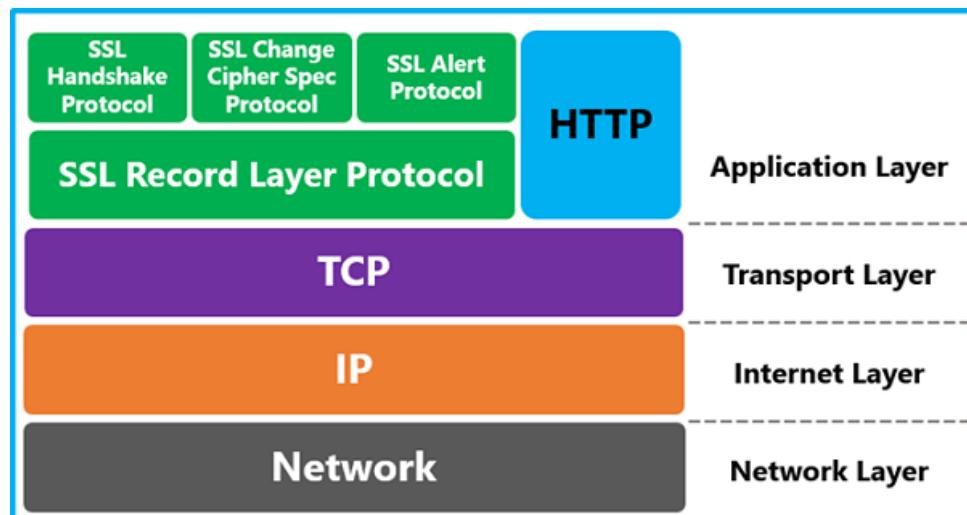
IPsec: Network Layer Security

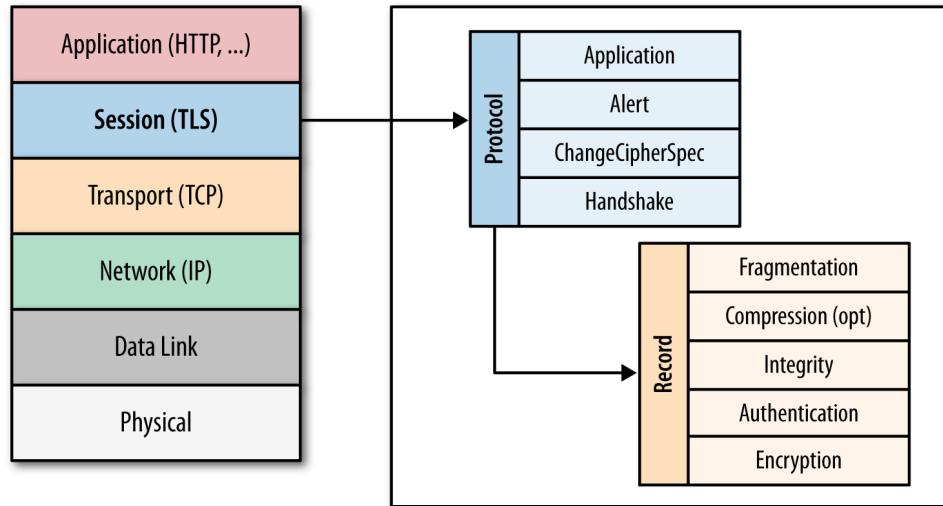
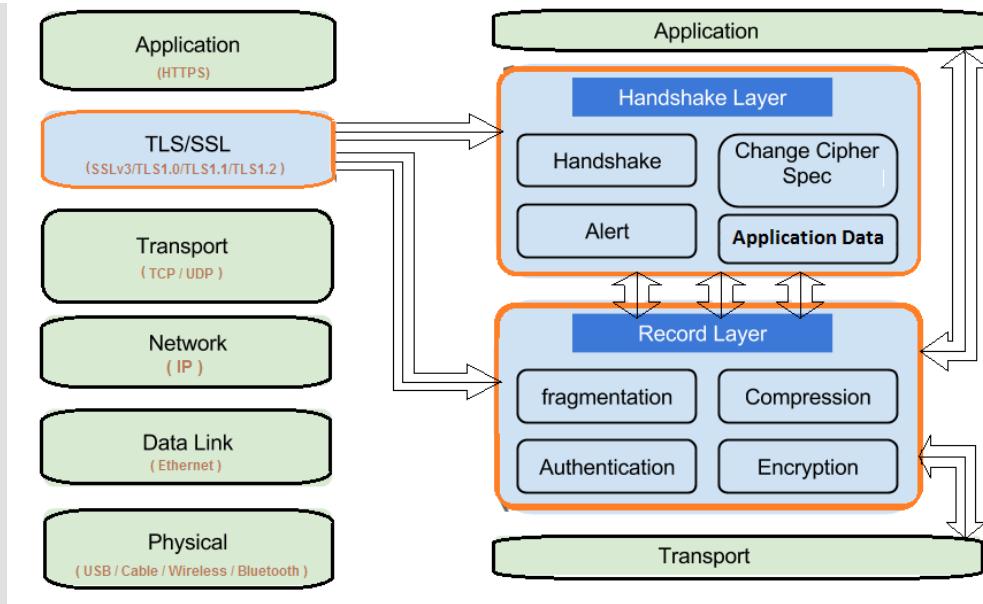
- **network-layer secrecy:**
 - sending host encrypts the data in IP datagram
 - TCP and UDP segments; ICMP and SNMP messages.
- **network-layer authentication**
 - destination host can authenticate source IP address
- **two principal protocols:**
 - authentication header (AH) protocol
 - encapsulation security payload (ESP) protocol
- **for both AH and ESP, source, destination handshake:**
 - create network-layer logical channel called a security association (SA)
- **each SA unidirectional.**
- **uniquely determined by:**
 - security protocol (AH or ESP)
 - source IP address
 - 32-bit connection ID

8: Network Security

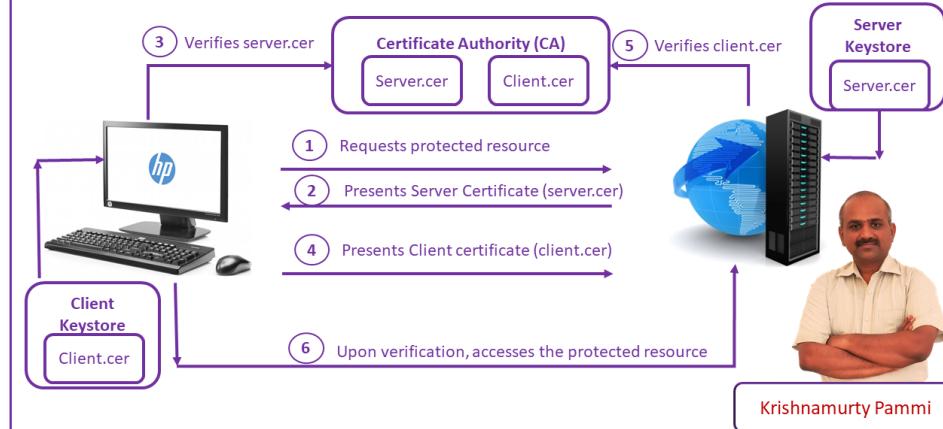
8-1

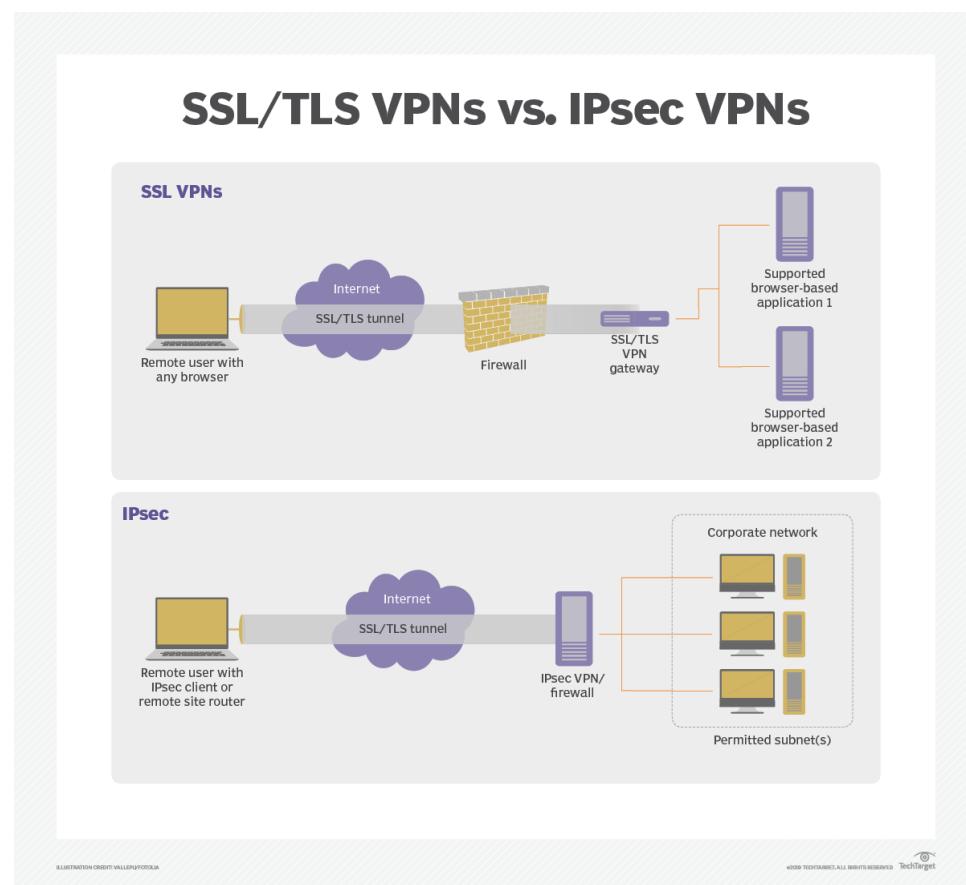
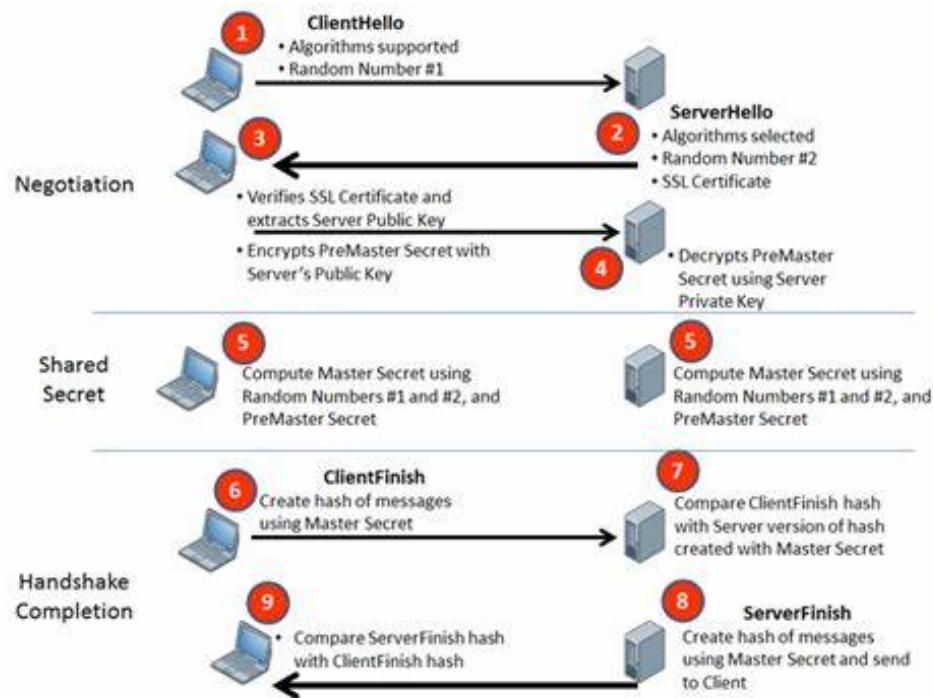
SSL(Secure Sockets Layers)

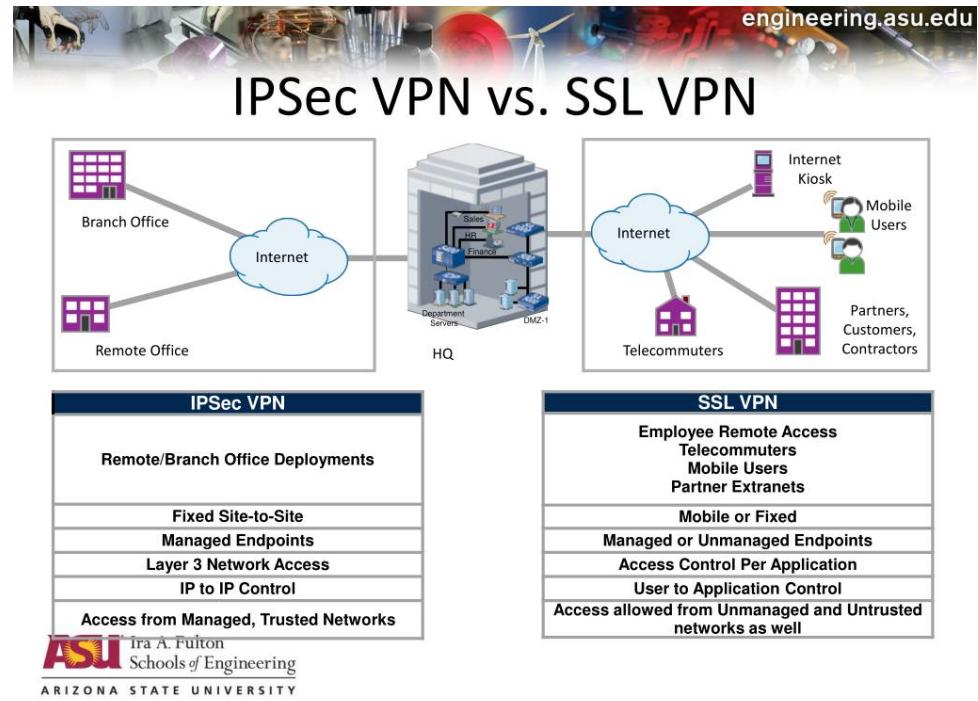




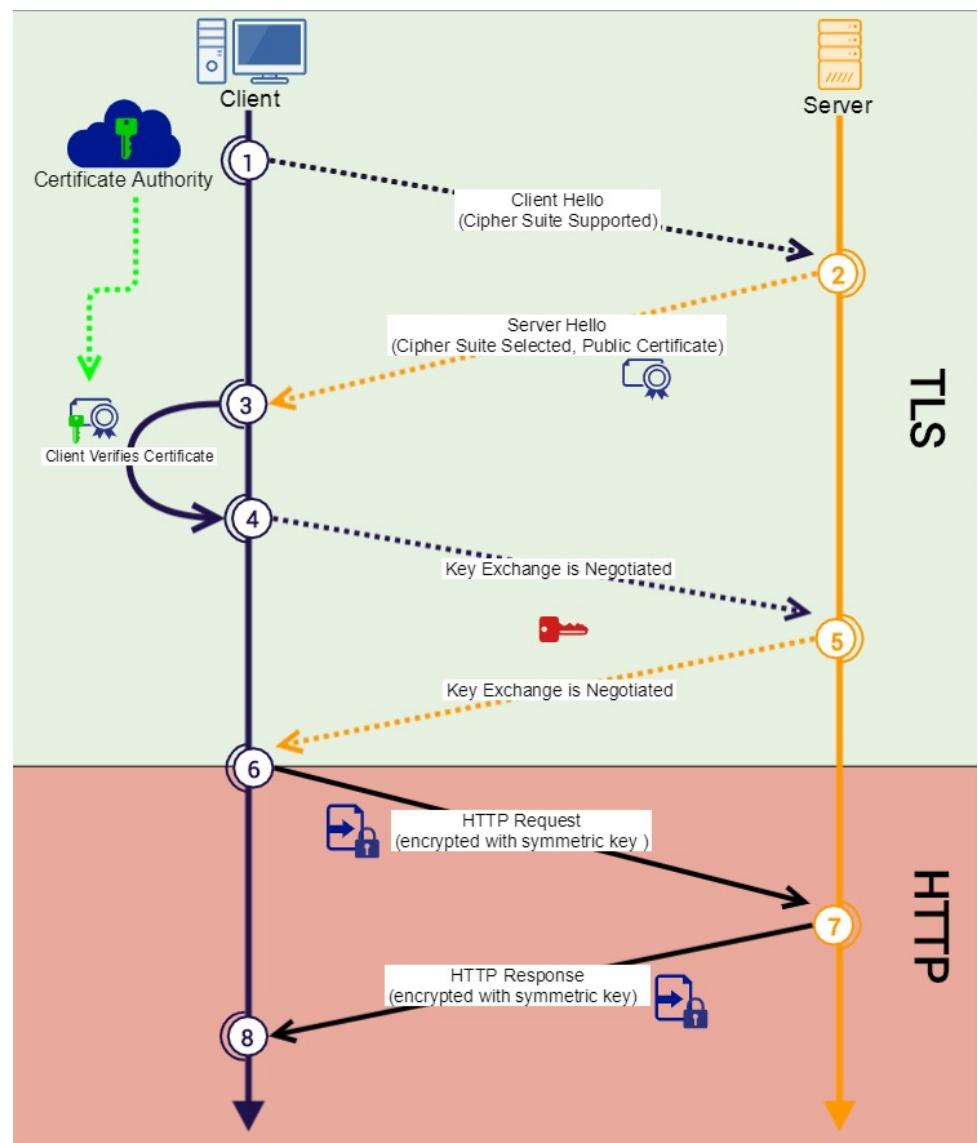
Web Security: Secure Socket Layer (SSL)



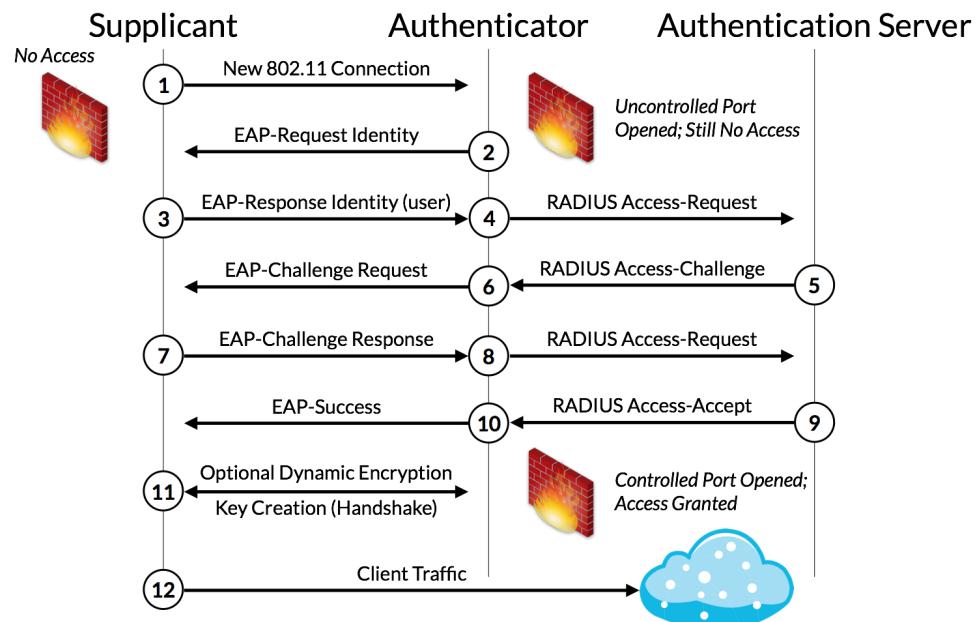




TLS



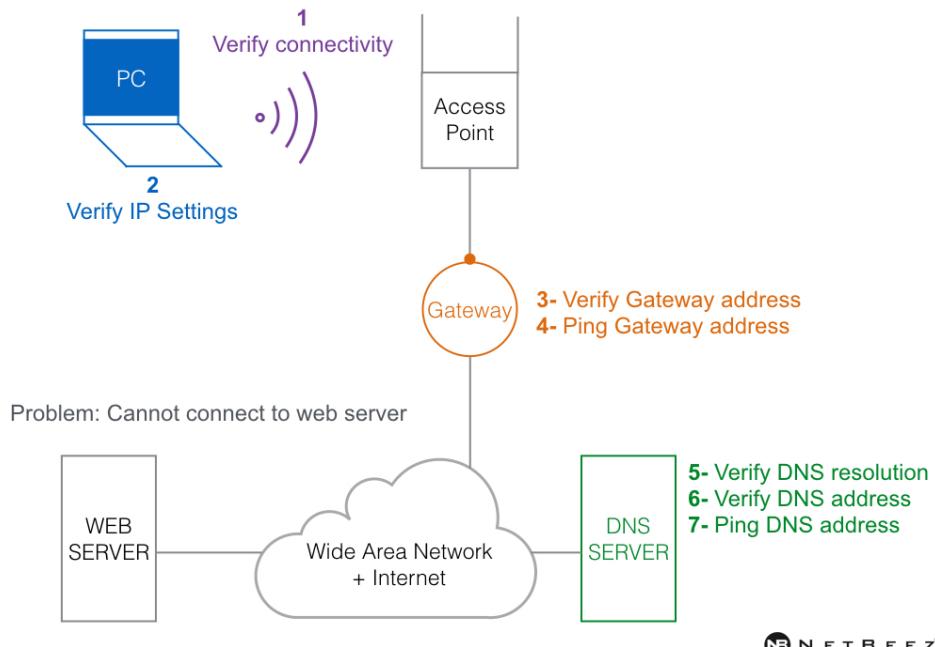
802.1X Authentication (EAP & RADIUS)



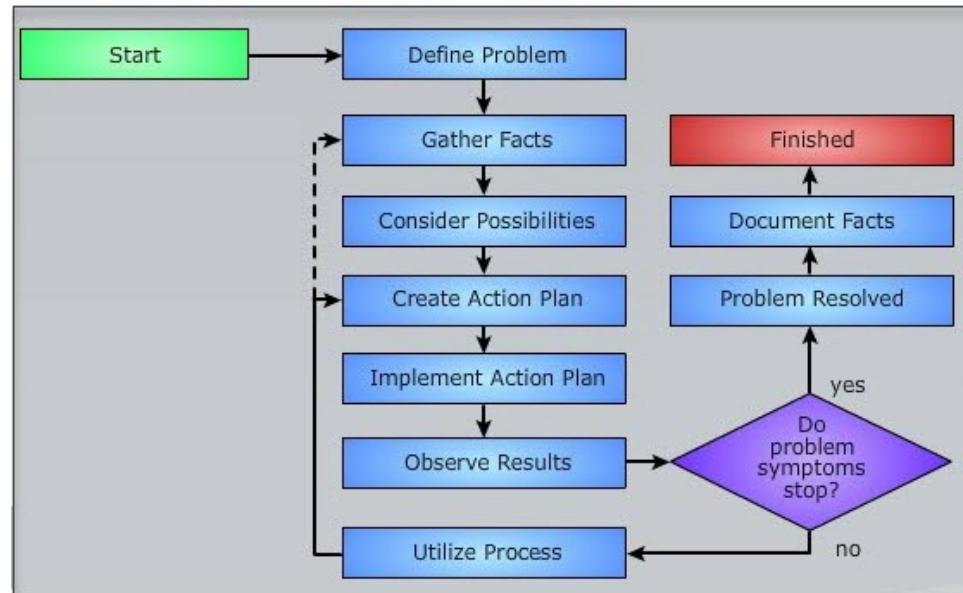
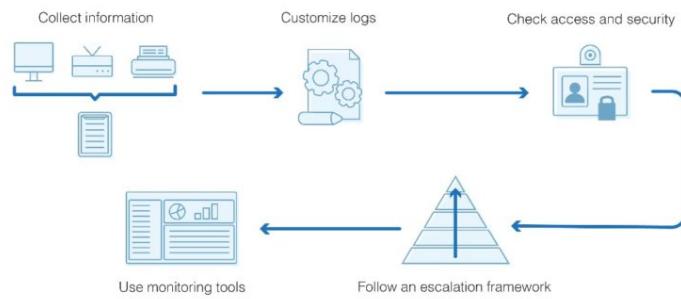
S S L V E R S U S T L S

SSL	TLS
Standard security protocol for establishing an encrypted link between a web server and a browser	Protocol that provides communication security between client/server applications that communicate with each other over the internet
Introduced in the year 1994 by Netscape Communications	Introduced in 1999 by Internet Engineering Task Force (IETF)
Stands for Secure Socket Layer	Stands for Transport Layer Security
Not as secure as TLS	More secure
Comparatively less complex	A complex protocol
Visit www.PEDIAA.com	

Network Troubleshooting



Network Troubleshooting Flowchart



Troubleshooting *STRATEGY*

1. Identify the symptoms and potential causes.

- ✓ Gather information about the problem.
- ✓ What is the problem?
- ✓ When did the problem occur?
- ✓ Specific error messages.
- ✓ Does the problem happen all the time or intermittently?

PowerCert

Troubleshooting *STRATEGY*

2. Identify the affected area.

- Is the problem isolated or spread across several locations?
 - If the problem affects everyone
 - ✓ Check the switch.
 - If the problem is isolated.
 - ✓ Check the individual cable.

PowerCert

Troubleshooting *STRATEGY*

3. Establish what has changed.

- ✓ Did anything change just prior to the problem happening?
- ✓ Was there any hardware removed or added?
- ✓ Was there any software installed or uninstalled?
- ✓ Was anything downloaded from the internet?

PowerCert

CompTIA Network+
PowerCert



Troubleshooting *STRATEGY*

4. Select the most probable cause.

- ✓ Look for simple solutions first.
- ✓ Does the device have power?
- ✓ Are the cables plugged in?
- ✓ Check the LEDs.

PowerCert



CompTIA Network+
PowerCert



Troubleshooting *STRATEGY*

5. Implement an action plan and solution including potential effects.

- ✓ The cautious phase.
- ✓ Must know what effect the action will have on the network.
- ✓ Will it affect the entire network or be isolated at one area?

PowerCert



CompTIA Network+
PowerCert



Troubleshooting *STRATEGY*

6. Test the result.

- ✓ Where you take action to solve the problem.
- ✓ Where you will know if your plan of action will solve the problem or not.

PowerCert



CompTIA Network+
PowerCert

Troubleshooting *STRATEGY*



7. Identify the results and effects of the solution.

- ✓ Has your plant of action solved the problem or not?
- ✓ What effect did it have on everyone else?
- ✓ Do the results show a temporary fix or a permanent one?

PowerCert

CompTIA Network+
PowerCert

Troubleshooting *STRATEGY*



8. Document the solution and process.

- ✓ Document the problem.
- ✓ Document what caused the problem.
- ✓ Document how the problem was fixed.



Network Administrator

PowerCert

-- Memo End --