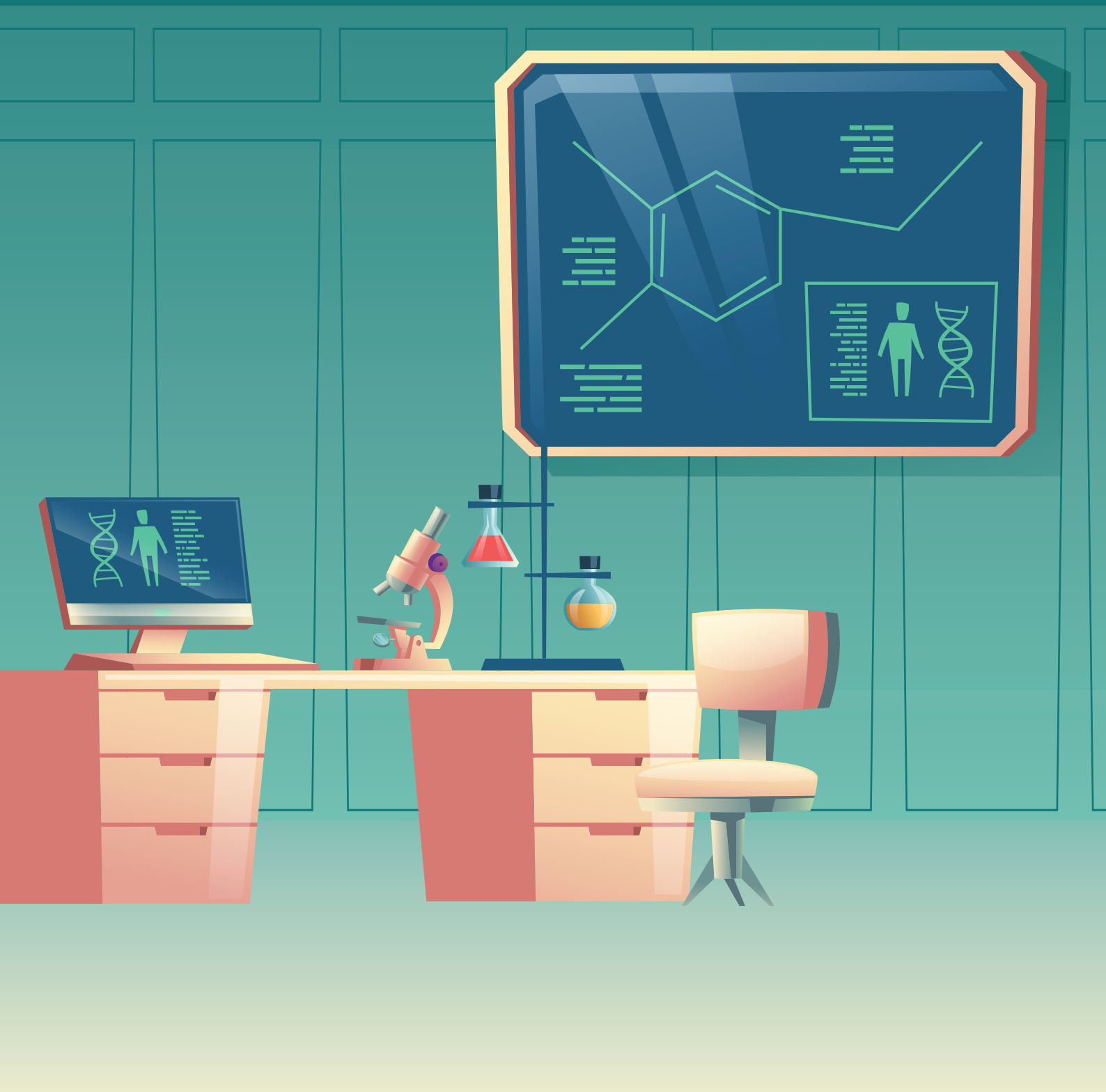
Laboratory Of Network Design and Configuration



FUNDAMENTALS OF IP NETWORKING

Internet network is composed by routers and host (divided in many other sub categories)

TCP/IP stack
- Application
- Transport (TCP/UDP)
- **Network (IP)** ← main focus
- Data link

Network layer: addressing and forwarding/routing

IP ROUTERS

Similar to a pc from hardware point of view with dedicated operating system (IOS for cisco devices). The IOS must be configured to properly work: IP addresses, routing protocols setting, security aspects etc.
A part from CPU, RAM, ROM etc main components are interfaces: LAN, WAN and console/AUX

LAN: connection toward a Local Area Network using ETherNet standard.
Depending on which we are connected to we have to use different kind of cable:
- crossover cable: connects devices of same type eg. host-to-host, switch-to-switch etc
- straight-through: connects devices of different type eg router-to-hub

Configuration of LAN interface can be done via TELNET protocol
WAN: Wide Area Network, no standards. Interfaces (Serial Ports) can be different since different technologies can be used.
AUX: used for router management (initial)

FUNCTIONS

Addressing. Forwarding. Fragmentation.

ADDRESSING

Identify each interface with an unique IP address. These addresses have a hierarchical structure IP = net_id + host_id (make use of Mask)

FORWARDING (routing)

Find the path from source to destination. Amazing. For this purpose the routing table is filled.

ROUTER CONFIGURATION

Configuration is done using CLI via console or telnet session. CLI has a hierarchical structure, different configuration levels: user EXEC mode, privileged EXEC mode (enable mode) and Global configuration mode.
In details:
① View only mode. Identified by ">" prompt. No configuration possible. "enable" command to access privileged mode
② Identified by "#" prompt. "exit" or "CTRL-Z" to come back to user mode (or "disable")
"configure terminal" to move to global configuration mode
③ Allow to configure interfaces, routing protocols etc. with specific prompt commands.

no ip domain lookup ⟶ avoid looking for command over the network
Change router name: hostname <name>

A password can be defined for different access way eg telnet.
Console port: "line console 0" ⟶ "password <password> ⟶ "login"
Telnet: "line vty 0 4" ⟶ "password <password> ⟶ "login"
↳ up to 5 telnet access possible

A password can be defined for enable mode in two ways:
① "enable password <password>"
② "enable secret <password>"
Difference is that in second case the password will be stored encrypted.

Running-config file store actual configuration of router. To save it (⟶ startup-config)
"copy running-config startup-config" (privileged mode at least)

INTERFACES

Serial Interfaces

"interface serial <0/0>" ⟶ "ip address <ip> <netmask>
If we are configuring interface responsable of clock: "clock rate <rate>"
Finally "no shutdown" to turn on interface IMPORTANT!

Ethernet Interfaces

Similar to previous. No clock required. "interface <e0>" ⟶ "ip address <ip> <mask>"
⟶ "no shutdown".

ROUTING

Routing table can be updated in two ways:
① Dynamic Routing: using info exchanged with different routers
② Static Routing: using configuration commands. Manual. Not scalable

Command for static routing: "ip route <dest_ip> <netmask> A / B
— A = output interface, only for point-to-point link. Or
— B = next hop router IP address.

Default route setup: "ip route 0.0.0.0 0.0.0.0 A/B"

Administrative Distance: measure of reliability of a route. The lower it is, the higher the reliability. Is possible to set it for a route adding it at the end of "ip route ..." command. Static routes are reliable by default (AD = 1).

"show ip route" → show routing table.

Connection between routers is done with serial interfaces (symbol on packet tracer). When adding the connection, the first router selected will be the one who provides the clock.

ROUTING PROTOCOLS

Protocols provide communication channel among routers to exchange reachability information about networks automatically.
Performance of a protocol is the convergence speed: time required so that all routers have same routing table. Wrong routing table may lead to packet loss.

AUTONOMOUS SYSTEM (AS): set of networks and routers managed by some administrator, seen from outside as a single entity (eg whole TIM network). Each AS will have its "own" routing protocol. Routing among ASes must be the result of an agreement among ASes

Routing protocols are divided in Distance Vector RP and Link State RP based on functioning principles or Interior Gateway Protocol and Exterior Gateway Protocol based on use scenario.

DISTANCE VECTOR ROUTING PROTOCOL

Each router only communicates with neighbor routers. Each of them notify its neighbors with its routing table periodically. Each row in the table is also characterized by a distance metric (depends on protocols eg. RIP protocol counts number of hops)
In this case, a router doesn't have a global network view. When receiving new routing table an algorithm updates local one if needed (eg best routes found)

RIP - ROUTING INFORMATION PROTOCOL

As said it use hops number as distance. Distance = 16 means unreachable. Updates every 30 secs. Administrative distance = 120
Two versions: RIPv1 (classful) and RIPv2 (classless). Main difference: RIPv1 does not send subnet mask during updates. Mask is chosen using classfull approach (A, B, C) or using the subnet mask of the network connected to the RIP manage incoming interface. Be careful when using.

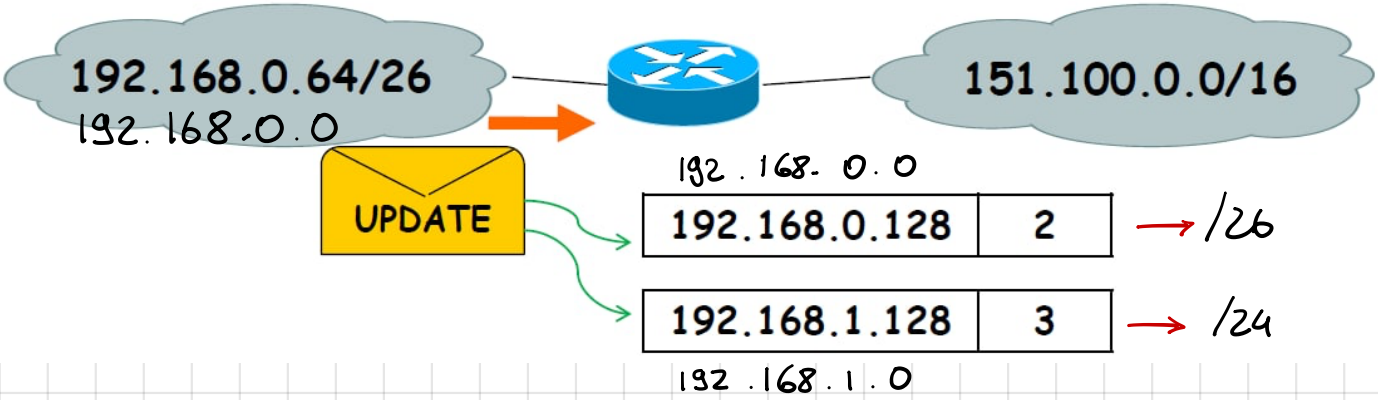
"router rip" → enable RIP
"network <net_add>" → define the interface running RIP, one for each directly connected network

If a route is not updated after "invalid timer" seconds it is flagged as invalid and distance set to 16. In the same way, after "flush timer" without updates it is removed from table

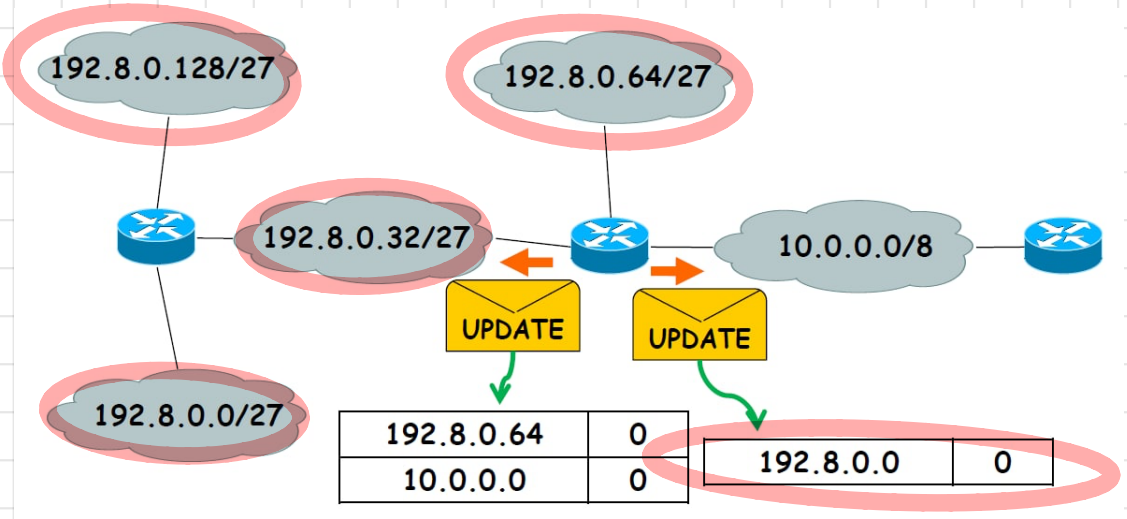
"passive-interface" → avoid sending RIP messages through a specific interface (eg LAN)

We said that in RIPv1 subnet mask of destination is not specified in RIP messages.
Lets introduce the Major Network concept: class A, B, C addresses are associated to an IP address eg major network of 192.168.0.128/25 is 192.168.0.0/24
i.e. A → /8 , B → /16 , C → /24

When a router receive an update message (RIP message) containg dest. IP it associates the subnet mask following two rules:
1) if destination network has a different major network than the one of the network directly connected to the receiving interface → use classful subnet mask
2) otherwise → use same subnet mask of the interface



When sending a RIP message instead, if the destination IP is obtained by subnetting: summarization
1) if dest address has different major network than the one directly connected to interface → use address of major network of "destination" (the one the message is coming from)
2) otherwise → use major network of interface



Problems:
- variable length mask not possible
- network with same major network must be contiguous.

RIPv2

Mask can be inserted in RIP message.
"router rip" → "version 2"
Anyway to fix previous problem we need to deactivate summarization → "no auto-summary"

In ALL protocols can be configured the default router on the BORDER router and then "distribute" this information
"default-information originate" (in border router)

LINK STATE - OPEN SHORTEST PATH FIRST (OSPF)

Every nodes knows whole network topology. Administrative distance is 110 → more reliable than RIP. Periodically each router send "Hello packet" to its neighbors to notify its status. Each of them describes its topology generating a Link State Advertisement (LSA) packet and flooding it over the network. In this way every router can compute shortest paths (Dijkstra algorithm) thanks to LSA database. (network topology is a weighted graph)

Transit Networks: more than one router
Stub Networks: connected to a single router

LSA

Composed by LS Header and LS Data. Type field in LSH identify the different LSAs: Router_LSA (type 1), Network_LSA (type 2) etc.

"router ospf <id>" - enable OSPF
<id> in range [1 - 65535], local validity, not related to topology (use what ever you prefer)

"network <network_add> <wildcard-mask> area <area-number>" → configure networks
wildcard-mask: invert 0s and 1s in netmask
area: AS can be divided in areas to solve scalability issue. Area 0 always present. Flooding is restricted to area.
"interface <interface>" → "ip ospf cost <cost>" → change interface cost for the path
"show ip ospf neighbor" → show neighbors...
"show ip ospf interface <interface>" → show cost of that interface

In Area Borde Router, the association between interfaces and areas must be specified:
"interface <interface>" → "ip ospf <ospf_id> area <area_id>

Network of different areas will be identified in routing table with "O IA"

Also in OSPF we can enable a passive interface

"default-information originate" → propagate default route with OSPF (ip route 0.0.0.0 0.0.0.0 int.)
OSPF must be enabled on private networks or public ones only, not both!

DYNAMIC HOST CONFIGURATION PROTOCOL (DHCP)

DHCP aims to configure hosts inside LAN, in particular ip address and subnet mask, default gateway and DNS server.

"ip dhcp pool <pool_name>" → start DHCP configuration
Pool name is the name used to identify DHCP configuration.
Next: "network <net_add> <netmask>" → assign dynamically address pool

To allow internet access we must configure default router and DNS server:
"default-router <router_add>"
"dns-server <server_add>"

Is possible to exclude some addresses from the pool to reserve them for some static server for example: "ip dhcp excluded-address <ip_add> (or range of IPs)"

Note: remove also IP you assigned statically

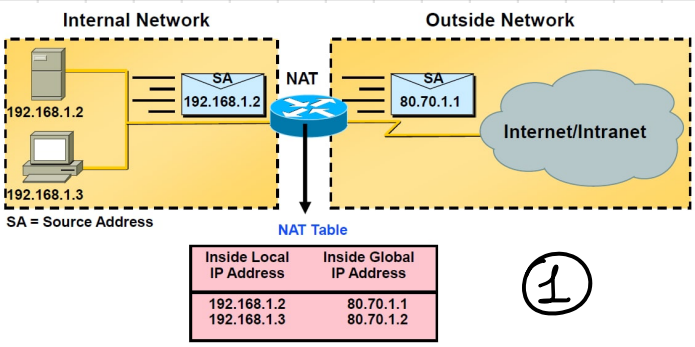
NETWORK ADDRESS TRANSLATION (NAT)

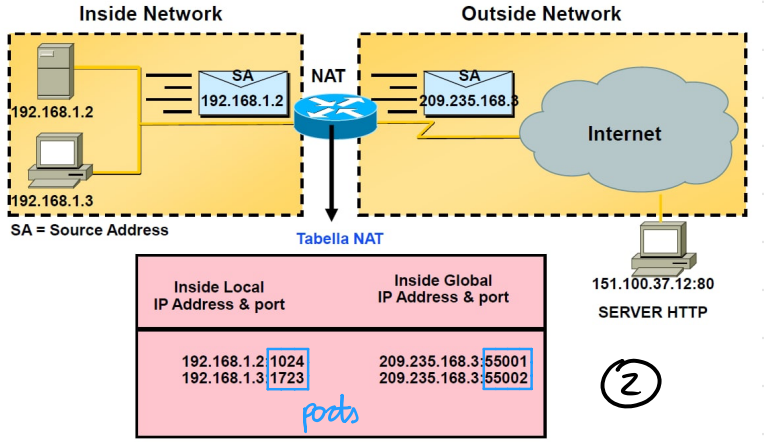
Problem: IPs are not enough to have unique IP on each host/device

Private IP: IP address which has validity only inside the network in which the device is. When "talking" outside the network, a "translation" will be used.
eg 10.0.0.0 (class A) or 192.168.0.0 - 192.168.255.0 (class C)

PRO: less public IP, real IP hidden
CONS: extra-load for access router and breaking of the layers separation rule.

There are two kinds of NAT:
1) Static: one-to-one translation, used on specific cases (no IP saving)
2) Dynamic: N-to-one translation





Entering interface configuration:
"ip nat inside" → set interface as inside interface
"ip nat outside" → " outside
} mandatory! Set who is inside (private) and who outside! (public)

- Static NAT:
  "ip nat inside source static <prv_add> <pub_add> → set translation rule (in global conf)
  Used for servers.
- Dynamic NAT
  Case 1: Pool of IPs, # of available public IP is lower than # hosts requiring internet
  - "access-list <access-list-number> permit <source_add> <wildcard>" → define private IP addresses
  - "ip nat pool <name> <start-ip> <end-ip> netmask <netmask>" → define pool of public IPs
  - "ip nat inside source list <acl-number> pool <name>" → translation rule

  Case 2: only available IP is the IP of outside interface (overload)
  - "access-list <access-list-number> permit <source_add> <wildcard>"
  - "ip nat inside source list <acl-number> interface <interface> overload

Note for Exercises: identify private networks by looking at their IP eg 10.0.0.0/8 = private
11.0.0.0/24 = public

Note: "access-list ..." command must be repeated as many times as many are private

networks "types" eg networks 10.1.0.5 and 10.10.5.7 → acc-list... 10.0.0.0 0.255.255.255

networks 192.168.10.0 and 192.168.10.3 → acc-list... 192.0.0.0 0.0.0.255

VIRTUAL LAN (VLAN)

Use of MAC addresses. MAC forwarding table is populated in the switch by means of a MAC learning procedure. The table "link" each known MAC to corresponding interface.
Switch IOS is similar to router's ones: ip, subnet, default gateway...

Use telnet connection: password must be configured to provide access and configuration
"interface vlan <number>" — enter interface conf., number 99 usually ↘ IP choosen in
"ip add ..." — "no shutdown" — as always management network range!
Then we must associate the virtual interface with the physical one(s)

VLAN i.e. is a way to create multiple networks on same physical infrastructure (network).

VLAN CONFIG

"vlan <number>" → define vlan and access its configuration "console", number = [1; 1005]
This command must be used on each switches "connected" to the defined vlan(s)

Access Port = interfaces connected to hosts belonging to a single VLAN
Trunk Port = interfaces connected to other switches/VLAN

"switchport mode access/trunk" → specify mode type
"switchport access vlan <number_of_vlan>" → specify the associated VLAN (only access port)
"switchport trunk native vlan <number> → specify a particular VLAN that "unify" all others during the transfer between switches, number whatever , same on both side of connection (only trunk port)

By default, trunk port forward and receives all VLANs frames.
Is possible to allow only a subset of them:
"switchport trunk allowed vlan <vlan_number_range>"

"show vlan brief" → on switch, show all VLAN configured, 1 if none configured.
"show <interface> switchport" → show mode used on interface

Management VLAN: used for management, usually VLAN 99. IPs of switches must be chosen among Management VLAN IPs.
To make it working good, telnet must be configured → add password
"line vty 0 15" → "password <pw>" → "login" (not privileged mode)
"enable password <pw>" (privileged, for remote configuration)

INTER-VLAN ROUTING

To let hosts on different VLANs to communicate a router is needed.
Two ways possible: traditional inter-VLAN and "Router-on-a-stick" inter-VLAN

TRADITIONAL INTER-VLAN

Router connected to the switch. Router must be connected to the switch with a number of physical links equal to the number of VLANs that will communicate (Access ports on switch side)
Default gateway of each host will be the address of router associated to its VLAN
Easy. But too many physical interfaces used

"ROUTER-ON-A-STICK" INTER-VLAN

Use a single physical interface split into virtual interfaces (equal to number of VLAN able to communicate)
Each subinterface is associated to a single VLAN (IP of VLAN block)
"interface <int>.<number>" → create subinterface, number whatever but use the same of desired VLAN
"encapsulation dot1q <vlan_number>" → assign VLAN to subinterface
"ip address ..." → set IP of subinterface ⟹ ADD to DEFAULT ROUTER in PCs!
Note: remember to "no shutdown" main interface
Trunk port on switch side

ACCESS CONTROL LIST (ACL)

List to control in/out traffic of a router, rules to determine if packets must be processed/forwarded or blocked/dropped.
Configured on router and applied to interfaces. Rules are defined on the basis of IP address, protocol, direction, port etc. Deny or Permit.
By default all packets are accepted/forwarded.
ACL can be applied to subinterfaces too!
Rules are checked in order → must be carefully defined


Bozon Zemo

3 types:
- Standard: packet filtering based on source IP address
- Extended: source/destination IP, protocol and destination port
- Named: can be both Standard or Extended but with more flexible configuration.

First two are identified by a unique (router-level) number.

STANDARD

"access-list <num> permit/deny <IP> <wildcard>" → define a rule (group)
In interface config: "ip access-group <num>" → associate ACL to interface
"host" keyword = 0.0.0.0 wildcard
"any" keyword = 255.255.255.255 wildcard
Identifier range [1; 99]

EXTENDED

Similar to previous one but with more filtering parameters.
Port number is optional and we can use: "eq" = equal, "neq" = not equal, "gt" = greater than and "lt" = less than
Identifier range [100; 199]
"access-list <num> permit/deny <protocol> <source> <src_mask> <dest> <dest_mask> <operator> <port_num>"
<protocol> → IP, TCP, UDP...
<operator> → eq, neq ... <port_num> → or telnet/ftp...

Protocol: using IP does no filtering since all packets are IP packets, TCP/UDP application level (with additional port field)
DNS = UDP, port 53
HTTP = TCP, port 80
TELNET = TCP, port 23/telnet
TFTP = UDP, port 69

NAMED

Can be standard or extended but can be modified without starting from scratch
"IP access-list standard/extended <name>" → create a named ACL and access specific CLI
"permit/deny <protocol> <source> ...." → configure the access-list as always.
Every rule inserted will have a number
"no <rule_num>" → delete a specific rule
Is possible to add a number before permit/deny to add the rule in a specific position.

Note: remember to apply ACL to subinterfaces in presence of VLANs, NOT to "main" interface!

To restrict telnet access:

① Create standard ACL permitting specific networks/host and denying everything else
"ip access-list standard <name>" → "permit A.B.C.D E.F.G.H"
→ Remember wildcard

② Configure telnet access
"line vty 0 4" (→ "password <pw>" → "login" if not done already)

③ Assign ACL to telnet
"access-class <name> in"

LAN SECURITY

Classic attack in a LAN is the MAC Address Flooding which exploits the security weakness of MAC forwarding table: frames with artificial source MAC address → saturated table → frames with new MAC destination addresses are forwarded in broadcast

Other attacks:

DHCP Spoofing, malicious DHCP server inserted in LAN → fake info notified to hosts. Man-in-the-middle.

DHCP Starvation, huge amount of requests to DHCP server to use all available IP addresses.

"ipconfig /all" → show PC MAC.

PORT SECURITY

Avoid MAC Address Flooding. Operation with single interface as target: limit end devices connected to it.

If a frame with MAC not allowed is received, interface goes to Violation Mode = shutdown

To configure, in interface command line: "switch port port-security" (in switch)

Then...

STATIC: statically configure allowed MACs
"switchport port-security mac-address <mac>"

DYNAMIC: allowed MACs are learned automatically, assign a max number of allowed addresses (1 by default), saves them on secure MAC table, a dedicated table
"switchport port-security maximum <num>" to increase max number

STICKY DYNAMIC: "switchport port-security mac-address sticky" save dinamically learned MACs in running config in addition to secure table

When in violation mode, interface must be "re-shutdown" and "re-turned on" by admin
"shutdown" → "no shutdown"

For auto-recovery: "errdisable recovery interval <time[30;86400]sec>"

"show port-security address" → in switch, to show MAC address table

DHCP SNOOPING

Avoid unauthorized DHCP server messages coming from an untrusted port = ports connected to host that shouldn't be providing DHCP server messages

"ip dhcp snooping" → enable snooping

"ip dhcp snooping vlan <VLAN_id>" → enable snooping on VLAN, at least one (id 1 if no VLANs)

"no ip dhcp snooping information option" → DO IT!

"ip dhcp snooping trust" → in interface command line, configure as trusted

PORT MIRRORING

A switch duplicate Ethernet frames to a second port connected to an Intrusion Detection System to analyze data.

SPAN is a type of port mirroring, Local in our case: mirroring on same switch.

SPAN session is an association between a source port (ingress, egress) and destination port (monitor port, only monitored traffic)

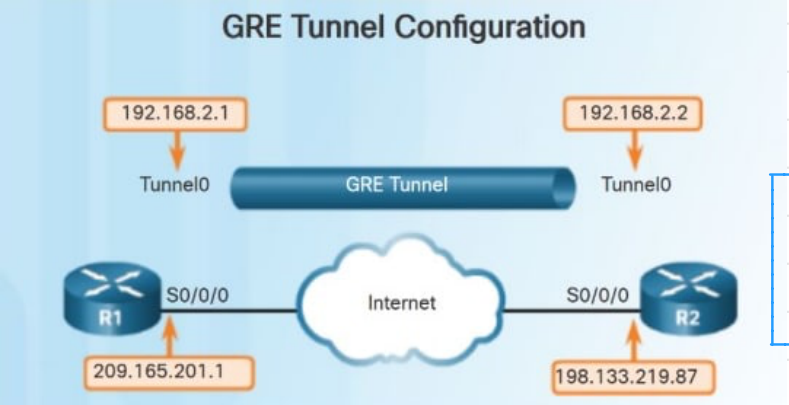
"monitor session <session_id> source interface <int1>" } duplicate packets from int1 to int2
"monitor session <session_id> destination interface <int2>" }

VIRTUAL PRIVATE NETWORK

Exploit tunneling mechanism to create a private network over a public one.
VPN Gateway is necessary (router or dedicated device)

Generic Routing Encapsulation - GRE

GRE is a non-secure site2site VPN tunneling protocol



"interface tunnel <0>" → create tunnel interface
"tunnel mode gre ip"
"ip address <192.168.2.1> <255.255.255.252>" → source IP
"tunnel source <S0/0/0>" → source port
"tunnel destination <198.133.219.87>" → dest pub. IP

→ Invent it

Note: add static route to connect (V)LANs connected by VPN
ip route <dest_network> <dest_mask> <tunnel_opposite_side-IP>

Final Notes

- DHCP works only after router interface IP address is assigned (subinterfaces if present)
- In presence of a single VLAN dont use trunk mode
- Internet Access = NAT → remember to set "not inside" on subinterfaces if present!
- ACL for VLAN not directly connected → use public IP of VLANS → differentiate IPs if needed
- Use IP for next hop mainly