Studing Cisco IOS commands, using recipes

by *luciano de falco alfano*

Version 0.1 on 18 Sep 2019 under license CC BY-SA 4.0.

Contents

Generalities....2
Target of this work and license....2
Structure of the document....2
Conventions....3
Procedures....3
All: basic configuration....3
Router: configuration....4
configure IPv4 direct connection....4
configure IPv4 loopback....4
execute a command....4
enable IPv6 unicast routing....4
reset to factory configuration....4
Router: static routing....4
set a static route....4
Router: dynamic routing....5
configure IPv4 RIP....5
what dynamic routing protocols does the router support?....6
Router, Switch: configuration....6
control user access....6
initial configuration and security....6
set what IOS to boot....7
show filtering....7
Switch: configuration....7
configure port at physical layer as auto-negotiation....7
configure port security....8
configure SSH....8
configure switch management interface (SVI)....8
rehabilitate a port in error disabled status....9
show switch configuration....9
Switch: VLAN....9
change port membership....9
configure port for Voip phone and PC....9
create VLAN and assign a port to it....10
configure trunk port....10
delete trunk port....11
delete VLAN....11
inter-VLAN routing: configure router-on-a-stick....11
restore to factory....12
Router: access control lists (ACL)....12
create named standard ACL and activate to an interface....12
create numbered standard ACL and activate to an interface....12
edit numbered standard ACL....12
remove ACL....13
restrict incoming connections to VTY using ACL....13
Router: dynamic host configuration protocol (DHCP)....13
DHCPv4 configure router as server....13
DHCPv4 configure router as relay....14
DHCPv4 configure router as client....14

DHCPv6 configure as stateless server....14
DHCPv6 configure router as stateful server....15
DHCPv6 configure router as client of stateless server....15
DHCPv6 configure router as client of stateful server....15
DHCPv6 configure router as relay agent....16
Router: NAT....16
IPv4 configure static NAT....16
IPv4 configure dynamic NAT....16
IPv4 configure port address translation (PAT)....17
IPv4 configure PAT, single address....17
Device management....18
Cisco discovery protocol (CDP)....18
Link layer discovery protocol (LLDP)....18
Set date and time....18
Syslog....19
File system....19
Backup and restore....19
Password recovery....20
Boot source....20
Licensing process....20
Other notions....21
powers of 2....21
Cidr and block....21
subnetting: steps....21
subnetting: example....21
variable length subnetting....22
Interfaces....22
Environments....22
notably addresses....23
IPv4....23
IPv6....23
IPV4 address schema....24
notably ports....25
Incoherences....25
Observations....25
Commands....26
List....26
Synthesis....29
References....31
Bibliography....31
Acronyms and abbreviations....31

Generalities

Target of this work and license

This documents intends to be a simplified reference manual of Cisco IOS. Author is writing it while studying a couple of courses of networking from the master degree in Computer Science at University of Camerino, Italy.

This is an ongoing work. So it isn't exhaustive nor complete.

You can use and modify this work under the terms of open source license CC BY-SA 4.0, without guarantee and under your responsibility.

Structure of the document

This document has these main chapters. The *Procedures* lists a bunch of useful operations: the recipes. *Other* notions has a set of notions to remember to use commands with success, or interpret their output. And finally the *Commands* part lists alphabetically the commands used in procedures.

Conventions

Command are preceded by a context showing the user prompt in CLI environment. So you know:

- the system type where to run the command: **R** is a router, **S** is a layer 2 switch;
- the environment where the command lives: if nothing then it is a base command, (config) in global configuration, (config-if) in interface configuration, and so on;
- and last, the user lever: **#** is an administrator, **>** if he's a normal user.

Highlighted text needs a check.

Commands are written in mono spaced font. Their parameters are in italic style. In examples (italic) parameters are to demo only, you'll need to adapt them to yours necessities.

Procedures

All: basic configuration

What configure? Hostname, banner, (five) passwords, interface descriptions.

```
01 Switch#configure terminal
(no lookup and set banner)
02 Switch(config)#no ip domain-lookup
03 Switch(config)#banner motd # don't enter, unless you are authorized
04 #
(set hostname)
05 Switch(config)#hostname SWF1
(password for privileged mode)
07 SWF1(config)#enable secret cisco
08 SWF1(config)#enable password cisco
(password for console)
10 SWF1(config)#line console 0
11 SWF1(config-line)#password cisco
12 SWF1(config-line)#login
13 SWF1(config-line)#exec-timeout 5 30
14 SWF1(config-line)#logging synchronous
(in case of switch, do: configure switch management interface (SVI))
(password for vty: telnet/ssh)
16 SWF1(config-line)#line vty 0 15
17 SWF1(config-line)#password cisco
18 SWF1(config-line)#login
19 SWF1(config-line)#exec-timeout 10 30
20 SWF1(config-line)#
(router only: password for aux)
22 RF1(config)#line aux 0
23 RF1(config-line)#password cisco
24 RF1(config-line)#login
25 RF1(config-line)#exec-timeout 10 45
26 RF1(config-line)#
```

Target.

Notes.

- Line 05: hostname by position (i.e. SWitch on Floor 1).
- line 03/04: the end of message (char #) is in a new line to avoid to glue more banners in a single message.
- Line 08: this only if you are using IOS < 10.3; otherwise use only enable secret …
- Line 11: don't use the same password everywhere (so: please don't do what I've done here!).
- Line 12: ever password+login.
- Line 13: timeout in 5 minutes and 30 seconds of inactivity.
- Line 14: don't mix messages from device with user input.

- Line 17: without set password, login function on VTYs is disabled (line 18 will raise an error for each VTY).
- Line 22…: only routers have aux port.

Router: configuration

configure IPv4 direct connection

```
R(config)# interface gigabitethernet 0/0
R(config-if)# description Link to LAN 1
R(config-if)# ip address 192.168.10.1 255.255.255.0
R(config-if)# no shutdown
```

Target.

Notes.

- Use `clock rate 128000` *in* case of DCE serial port.
- Use `ipv6` in case of IPv6 address (after Errore: sorgente del riferimento non trovata):

```
R(config-if)# ipv6 enable
R(config-if)# ipv6 address 2001:DB8:CAFE:3::1/64
```

configure IPv4 loopback

```
R(config)# interface loopback 0
R(config-if)# ip address 192.168.10.1 255.255.255.0
```

Target. Activate a loopback interface

Notes.

execute a command

```
R(config)# do show ip interface brief
```

Target. Execute a command from global configuration mode

Notes.

enable IPv6 unicast routing

```
R(config)# ipv6 unicast-routing
```

Target. Enable all the router's operations about IPv6. From setting IPv6 address to an interface, to routing IPV6 addresses.

Note.

reset to factory configuration

```
R# erase startup-config
R# reload
```

Target. Destroy the router's current startup configuration, resetting it to factory defaults.

Note. This is from packet tracer. In Cisco's online course, it seems to me could be different commands. Check the manufacturer documentation for your model of router.

Router: static routing

set a static route

```
R(config)# ip route 192.168.10.0 255.255.255.0 Serial0/0/0
R(config)# ip route 192.168.11.0 255.255.255.0 209.165.200.225
R(config)# ip route 0.0.0.0 0.0.0.0 Serial0/0/0
R(config)# ipv6 route ::/0 Serial0/0/0
(checking)
R# show ip route 192.168.10.0
R# show ip route
```

Target**.** Configure a static route. Check it showing the single route or the entire routing table. Use ipv6 (even for checking) in case of IPv6 net.

Notes**.**

- use `ipv6` in case of IPv6 route;
- 1st example is standard static route by exit interface;
- 2nd example is standard static route by next hop;
- 3rd example is a default static route (alias gateway of last resource) by exit interface; attention: IPv4 address and netmask aren't in italic style: they must be that values;
- 4th example is a default static route IPv6 by exit interface; as previous warning, IPv6 address must be that value.
- A summary static route is the aggregation of more adjoining nets, in one single bigger network; example: 172.20.0.0/16+172.21.0.0/16+172.22.0.0/16+172.23.0.0/16 → 172.20.0.0/14 because:

source networks	matching bits	maximum match	summary network
172.20.0.0/16	172. 0001 0100.-.-	172. 0001 01xx.-.-	172.20.0.0/14
172.20.0.0/16	172. 0001 0101.-.-		
172.20.0.0/16	172. 0001 0110.-.-		
172.20.0.0/16	172. 0001 0111.-.-		

- A floating static route is a backup route, usually not used; router uses it when main link goes down; use Administrative Distance to obtain this feature; an example, using summary address from previous point: if main link has AD value of 90 (EIGRP), set AD to 95 with this command[1]: `ip route` *`172.20.0.0 255.252.0.0 S0/1/0 95`*.

Router: dynamic routing

configure IPv4 RIP

```
R(config)# interface g0/0
R(config-if)# ip address 192.168.10.1 255.255.255.0
R(config-if)# interface s0/0/0
R(config-if)# ip address 209.165.200.225 255.255.255.252
R(config-if)# no shutdown
R(config-if)# interface s0/0/1
R(config-if)# ip address 209.165.200.233 255.255.255.252
R(config-if)# no shutdown
R(config-if)# exit
R(config)# ip route 0.0.0.0 0.0.0.0 s0/0/1
(configure RIP)
R(config)# router rip
R(config-router)# version 2
R(config-router)# no auto-summary
R(config-router)# network 192.168.10.0
R(config-router)# passive-interface g0/0
R(config-router)# network 209.165.200.224
R(config-router)# network 209.165.200.232
R(config-router)# default-information originate
```

Target. Configure RIPv2 dynamic routing protocol. The greyed part is preparatory: that commands configure the Gigabit Ethernet interface and two serial interfaces. The serials are /30 nets (two useful addresses each). And the s0/0/1 is even a gateway of last resort (all unresolved addresses are forwarded here).

Note.

- force version 2;
- disable auto summary, so we can work with classless networks;

1 Make attention to the last parameter: it's the AD.

- example of passive interface (no routers on 192.168.10.0);
- two classless networks;
- propagate gateway of last resort.
- **Warning**: IPv6 dynamic routing configuration is completely different.

what dynamic routing protocols does the router support?

```
R(config)# router ?
```

Target.

Note.

- be aware of global configuration environment, and
- missing the indication about `ip` before `router`;
- use `ipv6 router ?` in case of IPv6 routing.

Router, Switch: configuration

control user access

```
RS(config)# service password-encryption
RS(config)# login block-for 120 attempts 3 within 60
RS(config)# security password min-length 8
(enable ssh)
RS# show ip ssh
RS# configure terminal
RS(config)# ip domain-name ldfa.org
RS(config)# crypto key generate rsa general-keys modulus 1024
RS(config)# ip ssh version 2
RS(config)# username luciano privilege 15 secret pass1234*
(secure lines ... console)
RS(config)# line console 0
RS(config-line)# password pass4321*
RS(config-line)# login
(... vty)
RS(config-line)# line vty 0 15
RS(config-line)# login local
RS(config-line)# transport input ssh
RS(config-line)# exec timeout 10
(... aux)
RS(config-line)# line aux 0
RS(config-line)# login local
RS(config-line)# transport input ssh
RS(config-line)# exec timeout 10
(set banner)
RS(config)# banner motd delimiter message delimiter
```

Target. Secure access to intermediate device, using ssh as protocol.

Note. These are the logical macro steps:

- enable password-encryption
- enable check on login trails: if too many fails block login for 120 seconds;
- minimum length of password
- enable ssh (check, generate keys, create user)
- secure lines: console, vty and aux
- set banner

initial configuration and security

```
RS> enable
RS# configure terminal
RS(config)# no ip domain-lookup
```

```
RS(config)# hostname R1
R1(config)# ip domain-name ldfa.it
(restrict privileged mode)
R1(config)# enable secret pass1234*
(restrict access to user ...)
(... general attributes)
R1(config)# service password-encryption
R1(config)# login block-for 120 attempts 3 within 60
R1(config)# security password min-length 8
(... enable ssh)
R1# show ip ssh
R1(config)# ip domain-name ldfa.org
R1(config)# crypto key generate rsa general-keys modulus 1024
R1(config)# ip ssh version 2
(... configure local user, top admin privilege)
R1(config)# username luciano privilege 15 secret pass1234*
(... protect console)
R1(config)# line console 0
R1(config-line)# password pass4321*
R1(config-line)# login
(... protect vtys)
R1(config-line)# line vty 0 15
R1(config-line)# password pass4321*
R1(config-line)# login local
R1(config-line)# transport input ssh
R1(config-line)# exec timeout 10
(... protect aux)
R1(config-line)# line aux 0
R1(config-line)# password pass4321*
R1(config-line)# login local
R1(config-line)# transport input ssh
R1(config-line)# exec timeout 10
```

Target. Initial configuration with security.

Note.

- Initial: disable NDS lookup, assign hostname, assign domain name;
- then security actions.

set what IOS to boot

```
RS(config)# boot system flash:/c2960-lanbasek9-mz.150-2.SE/c2960-lanbasek9-mz.150-2.SE.bin
```

Target. Set the BOOT environment variable, used by the boot loader, to indicate where to find the IOS to load.

Note.

show filtering

```
RS# show ... | {section | include | exclude | begin} a_filter
```

Target. Filter show output using the filter on right side of the pipe sign.

Note. Packet Tracer doesn't support the **section** keyword.

Switch: configuration

configure port at physical layer as auto-negotiation

```
S(config)# interface fa0/1
S(config-if)# duplex auto
```

```
S(config-if)# speed auto
S(config-if)# mdix auto
```

Target. Set physical port configuration as auto-negotiation

Note.

configure port security

```
S(config-if)# switchport mode access
S(config-if)# switchport port-security
(configure port)
S(config-if)# switchport port-security mac-address 0A00.2700.0016
S(config-if)# switchport port-security mac-address sticky
S(config-if)# switchport port-security violation protect
(checking port security)
S(config-if)# end
S# show port-security interface fa0/1
S# show port-security address
S# show port-security
```

Target. Set port security with allowed MAC addresses. The first two lines prepare the port: port-security requires an access (or trunk) port and it must be enabled (the 2nd line). The last three lines are about to check the configuration of the port.

Note.

- 1st example is a static secure MAC addresses;
- 2nd example is a sticky secure MAC address; usually preceded by `switchport port-security maximum max`, where max is the maximum number of learned MAC addresses
- 3rd example is about setting the violation mode.

configure SSH

```
(do All: basic configuration)
S(config)# do show ip ssh
S(config)# ip domain-name ldfa.org
S(config)# crypto key generate rsa general-keys modulus 1024
S(config)# ip ssh version 2
(create local user as administrator)
S(config)# username luciano privilege 15 secret pass1234*
(force to use ssh on vty)
S(config-line)# line vty 0 15
S(config-line)# login local
S(config-line)# transport input ssh
S(config-line)# exec timeout 10
```

Target. Force to use ssh protocol on VTY.

Note.

configure switch management interface (SVI)

We need to know these basic concepts about remote management of switches. It means:

- a SVI interface with an IP address;
- vty lines right configured to use the SVI interface to accept telnet (or SSH) terminal connection; right configured means to be able to do login, but to be able to do login, before we had setup a password for these vty;
- PC to use to do telnet to the switch needs an IP address.

Here we show only the 1st point. To see a more complete configuration (that refers even this configuration) check: All: basic configuration.

```
S(config)# vlan 99
S(config-vlan)# exit
S(config)# interface vlan 99
S(config-if)# ip address 192.168.10.2 255.255.255.0
S(config-if)# no shutdown
```

```
S(config-if)# exit
S(config)# ip default-gateway 192.168.10.1
S(config)# interface f0/1
S(config-if)# switchport mode access
S(config-if)# switchport access vlan 99
```

Target. Manage the switch using an inbound remote terminal on port *f0/1*.

Note. Due to security reason we use a VLAN different from the default (which is VLAN 1). So:

- create a new VLAN (id 99)
- name it
- set SVI on VLAN 99 with IPv4 address and `no shutdown` command
- set default-gateway IPv4 address (only if we need to manage the switch from another LAN)
- assign port fa0/1 to VLAN 99.

  Note some disadvantages. Security is safe, but we are locked to an explicit fixed port to manage the switch. Not the top of comfort. If you are a little masochist, you can fix even the MAC address of the PC to use enabling the port security (see: configure port security).

  Otherwise you can assign all ports to VLAN 99. But it will become a problem to manage multiple VLANS: a port can bind only a single VLAN as data access.

rehabilitate a port in error disabled status

```
S(config)# interface fa0/1
S(config-if)# shutdown
S(config-if)# no shutdown
```

Target. To enable again a port where there was a security violation.

Note.

show switch configuration

```
S# show running-config
S# show startup-config
S# show interface vlan1
S# show ip interface vlan1
S# show version
S# show interface f0/1
S# show flash
```

Target. To analyse the switch configuration.

Note. The `show flash` command could be `dir flash:`

Switch: VLAN

change port membership

```
S(config)# interface fa0/18
S(config-if)# no switchport access vlan
S(config-if)# switchport access vlan 20
```

Target. Change port membership from one VLAN to another.

Note.

- 2nd line set membership to VLAN 1; observe that we don't need to indicate previous VLAN id;
- 3rd line change membership to VLAN 20, whatever was the previous id.

configure port for Voip phone and PC

```
S(config)# cdp run
S(config)# vlan 20
S(config-vlan)# name STUDENT
S(config)# vlan 150
S(config-vlan)# name VOICE
S(config-vlan)# exit
(configure for voip and data)
```

```
S(config)# interface fa0/18
S(config-if)# switchport mode access
S(config-if)# switchport access vlan 20
S(config-if)# mls qos trust cos
S(config-if)# switchport voice vlan 150
```

Target. Use port *fa0/18* to connect a Cisco Voip phone to switch, and a PC to the phone. It needs two VLAN: a voice VLAN for Voip phone and a data VLAN for PC. This is the only case for a port to bind more than one VLAN, otherwise it's a trunk.

Note.

- Command `switchport voice vlan` needs Cisco Discovery Protocol (see ref.3), so the 1st line runs it to be sure it's up.
- Prepare two VLANs: 20 and 150;
- then configure fa0/18 to stick both of them;
- command `mls quos trust cos` enable Quality of Service (qos) driven by the Class of Service (cos) of the Voip phone;
- command `switchport voice vlan` configure automatically the port like a mini trunk, with two VLANs: the voice and the data (see ref.3).

create VLAN and assign a port to it

```
S(config)# vlan 20
S(config-vlan)# name STUDENT
S(config-vlan)# exit
S(config)# interface range fa0/2-10
S(config-if-range)# switchport mode access
S(config-if-range)# switchport access vlan 20
S(config-if-range)# end
(checking)
S# show vlan brief
S#show vlan brief
VLAN Name                             Status    Ports
---- -------------------------------- --------- -------------------------------
1    default                          active    Fa0/21, Fa0/22, Fa0/23, Fa0/24
                                                Gig0/1, Gig0/2
20   STUDENT                          active    Fa0/2, Fa0/3, Fa0/4, Fa0/5
                                                Fa0/6, Fa0/7, Fa0/8, Fa0/9
                                                Fa0/10
...
S# show interfaces fa0/2 switchport
Name: Fa0/2
Switchport: Enabled
Administrative Mode: static access
Operational Mode: static access
Administrative Trunking Encapsulation: dot1q
Operational Trunking Encapsulation: native
Negotiation of Trunking: Off
Access Mode VLAN: 20 (STUDENT)
Trunking Native Mode VLAN: 1 (default)
Voice VLAN: none
...
```

Target. Create vlan 20, named STUDENT and assign ports from fa0/2 to fa0/10 to it.

Note.

configure trunk port

```
S(config)# VLAN 20
S(config-vlan)# name STUDENT
S(config)# VLAN 30
S(config-vlan)# name PROFESSOR
S(config)# VLAN 99
S(config-vlan)# name NATIVE
```

```
(configure trunk port)
S(config)# interface fa0/1
S(config-if)# switchport mode trunk
S(config-if)# switchport trunk native vlan 99
S(config-if)# switchport trunk allowed vlan 20,20,99
(checking)
S(config-if)# end
S# show interfaces f0/1 switchport
```

Target. Configure a port as trunk.

Note. The native VLAN must match to both ends of the trunk. It seems a PC on a native port cannot ping to a PC on other VLAN even if right inter routed.

delete trunk port

```
S(config)# interface fa0/1
S(config-if)# no switchport trunk allowed vlan
S(config-if)# no switchport trunk native vlan
S(config-if)# switchport mode access
(checking)
S(config-if)# end
S# show interfaces f0/1 switchport
```

Target. Configure a port as trunk.

Note.

- The native VLAN must match to both ends of the trunk.
- Line 4 force port mode to access. Just to be sure.

delete VLAN

```
(move ports to default VLAN)
S(config)# interface range fa0/2-10
S(config-if-range)# no switchport access vlan
S(config-if-range)# end
(delete VLAN)
S(config)# no vlan 20
```

Target. Delete a VLAN.

Note. Before move its ports to another VLAN for example the default VLAN.

inter-VLAN routing: configure router-on-a-stick

```
(configure switch trunk port on switch)
S(config)#interface g0/1
S(config-if)#switchport mode trunk
S(config-if)#switchport trunk native vlan 99
S(config-if)#switchport trunk allowed vlan 20,30,99
S(config-if)#end
S#show interfaces g0/1 switchport
... Administrative Mode: trunk ...
Trunking Native Mode VLAN: 99 (accesso-remoto) ...
Trunking VLANs Enabled: 20,30,99 ...
(configure router interface)
R(config)#interface g0/0.20
R(config-subif)#encapsulation dot1q 20
R(config-subif)#ip address 172.17.20.254 255.255.255.0
R(config-subif)#interface g0/0.30
R(config-subif)#encapsulation dot1q 30
R(config-subif)#ip address 172.17.30.254 255.255.255.0
R(config-subif)#interface g0/0.99
R(config-subif)#encapsulation dot1q 99 native
R(config-subif)#ip address 172.17.99.254 255.255.255.0
R(config-subif)#interface g0/0
R(config-if)#no shutdown
(checking)
```

```
R(config-if)#end
R#show ip route
```

Target. Create routes among VLANs, using a single interface on router.

Note. PC on native VLAN seems it doesn't speak with PCs on non native VLANs.

restore to factory

```
S# delete flash:vlan.dat
S# erase startup-config
```

Target. Delete all VLAN and startup configurations.

Note.

Router: access control lists (ACL)

Remember: there is always an implicit `deny any` at the end of an ACL that blocks everything.

create named standard ACL and activate to an interface

```
(creating)
S(config)# ip access-list standard P_LAN_10
S(config-std-nacl)# deny host  192.168.10.11
S(config-std-nacl)# permit  192.168.10.0  0.0.0.255
S(confgi-std-nacl)#exit
(activating)
S(config)# interface s0/0/0
S(config-if)# ip access-group P_LAN_10 out
(checking)
S# show access-list
Standard IP access list 10
    10 deny 192.168.10.11 0.0.0.0
    20 permit 192.168.10.0 0.0.0.255
S# show ip interface s0/0/0
... Outgoing access list is 10 ...
```

Target. Create standard ACL 10 and apply it to interface s0/0/0

Note.

- This ACL permit hosts from 192.168.10.0/24 LAN except 192.168.10.11.

create numbered standard ACL and activate to an interface

```
(creating)
S(config)# access-list 10 remark permit hosts from the 192.168.10.0/24 LAN except ...11
S(config)# access-list 10 deny host  192.168.10.11
S(config)# access-list 10 permit  192.168.10.0  0.0.0.255
(activating)
S(config)# interface s0/0/0
S(config-if)# ip access-group 10 out
(checking)
S# show access-list
Standard IP access list 10
    10 deny 192.168.10.11 0.0.0.0
    20 permit 192.168.10.0 0.0.0.255
S# show ip interface s0/0/0
... Outgoing access list is 10 ...
```

Target. Create standard ACL 10 and apply it to interface s0/0/0.

Note. This ACL permit hosts from 192.168.10.0/24 LAN except 192.168.10.11.

edit numbered standard ACL

```
R# show access-lists
Standard IP access list 10
    10 deny host 192.168.10.99
    20 permit 192.168.10.0 0.0.0.255
```

```
(editing)
R# conf t
R(config)# ip access-list standard 10
R(config-std-nacl)# no 10
R(config-std-nacl)# 10 deny host 192.168.10.11
(checking)
R(config-std-nacl)# end
R# show access-lists
Standard IP access list 10
    10 deny host 192.168.10.11
    20 permit 192.168.10.0 0.0.0.255
```

Target. Modify wrong Access Control Entry of ACL 10. Correct denied host must be 192.168.10.11, not ...99.
Note.

- 1st part checks ACE sequence numbers using `show access-list` command.
- 2nd part makes edits:
  - `no 10` removes the wrong line, then
  - `10 deny host 192.168.10.11` rewrites the correct entry.
- Renaming a named ACL is the same thing. Only use `ip access-list standard acl-name` command.

remove ACL

```
R(config)# access-list 10 permit 192.168.10.0 0.0.0.255
R(config)# interface s0/0/0
R(config-if)# ip access-group 10 out
(removing)
R(config-if)# no ip access-group 10 out
R(config)# exit
R(config)# no access-list 10
(checking)
R# show ip interface s0/0/0
...
```

Target. Removing standard ACL 10.
Note. It is necessary remove manually all the application of an ACL before to remove it.

restrict incoming connections to VTY using ACL

```
R(config)# access-list 21 permit 192.168.10.0 0.0.0.255
R(config)# access-list 21 deny any
(restricting)
R(config)# line vty 0 15
R(config-line)# login local
R(config-line)# transport input ssh
R(config-line)# access-class 21 in
R(config-line)# end
(checking)
R# clear access-list counters
connect from 192.168.10.10, it should work
then try to connect from 192.168.11.11, it shouldn't work
R# show access-list
```

Target. Protect VTY connection to router restricting access to it using ACL.
Note.

Router: dynamic host configuration protocol (DHCP)

Cisco routers can perform the DHCP service.

DHCPv4 configure router as server

```
(defining excluded addresses)
R(config)# ip dhcp excluded-address 192.168.10.1 192.168.10.9
```

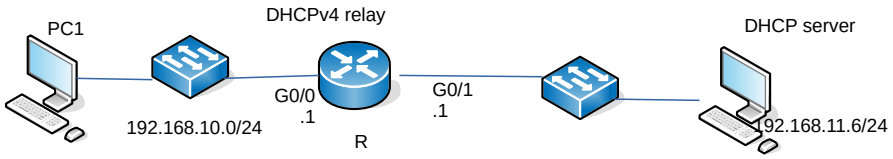
```
R(config)# ip dhcp excluded-address 192.168.10.254
(defining dhcp pool)
R(config)# ip dhcp pool POOL-1
R(dhcp-config)# network 192.168.10.0 255.255.255.0
R(dhcp-config)# default-router 192.168.10.1
R(dhcp-config)# dns-server 192.168.11.5
R(dhcp-config)# domain-name dhcp-example.com
(checking)
R(dhcp-config)# end
R# show running-config | section dhcp
...
```

Target. Set a base DHCPv4 server

Note.

- 1st: exclude static addresses (servers, internetwork devices, …);
- then define the DHCP pool (network, default gateway, DNS server and domain name).

DHCPv4 configure router as relay



Picture 1: dhcp relay agent schema

```
R(config)# interface g0/0
R(config-if)# ip helper-address 192.168.11.6
(checking)
R(config-if)# end
R# show ip interface g0/0
...
    Helper address is 192.168.11.6
```

Target. Given the illustrated previous topology. PC1 is a DHCPv4 client on 192.168.10.0/24 network. But DHCPv4 server is on 192.16.11.0/24 network. PC1 cannot reach the DHCP server because R, as a router, cannot forward broadcast packets. So we configure R as a DHCP relay to the DHCP server.

Note.

DHCPv4 configure router as client

```
R(config)# interface g0/0
R(config-if)# ip address dhcp
R(config-if)# no shutdown
(checking)
R(config-if)# end
R# show ip interface g0/0
...
    Helper address is 192.168.11.6
```

Target. Not a usual configuration. To try id a DHCP server is runnig without using a PC.

Note.

DHCPv6 configure as stateless server

```
R(config)# ipv6 unicast-routing
(configuring)
R(config)# ipv6 dhcp pool IPV6-STATELESS
R(config-dhcpv6)# domain-name example.com
R(config-dhcpv6)# exit
R(config)# interface g0/1
R(config-if)# ipv6 address 2001:db8:cafe:1::1/64
```

```
R(config-if)# ipv6 dhcp server IPV6-STATELESS
R(config-if)# ipv6 nd other-config-flag
(checking)
R(config-if)# end
R# show ipv6 dhcp pool
...
R# show ipv6 interface g0/1
...
R# debug ipv6 dhcp detail
...
```

Target. Configure stateless DHCPv6 server.

Note. Here there aren't excluded addresses because server tell to the client only its network address (prefix and prefix length).

DHCPv6 configure router as stateful server

```
R(config)# ipv6 unicast-routing
(configuring)
R(config)# ipv6 dhcp pool IPV6-STATEFUL
R(config-dhcpv6)# address prefix 2001:db8:cafe:1::/64 lifetime infinite
R(config-dhcpv6)# dns-server  2001:db8:cafe:aaaa::5
R(config-dhcpv6)# domain-name example.com
R(config-dhcpv6)# exit
R(config)# interface g0/1
R(config-if)# ipv6 address 2001:db8:cafe:1::1/64
R(config-if)# ipv6 dhcp server IPV6-STATEFUL
R(config-if)# ipv6 nd managed-config-flag
(checking)
R(config-if)# end
R# show ipv6 dhcp pool
...
R# show ipv6 dhcp binding
...
R# show ipv6 interface g0/1
...
```

Target. Configure stateless DHCPv6 server.

Note. Here there aren't excluded addresses because server tell to the client only its network address (prefix and prefix length)

DHCPv6 configure router as client of stateless server

```
R(config)# ipv6 unicast-routing
(configuring)
R(config)# interface g0/1
R(config-if)# ipv6 enable
R(config-if)# ipv6 address autoconfig
(checking)
R(config-if)# end
R# show ipv6 interface g0/1
...
```

Target. Configure DHCPv6 client to test server.

Note.

DHCPv6 configure router as client of stateful server

```
R(config)# ipv6 unicast-routing
(configuring)
R(config)# interface g0/1
R(config-if)# ipv6 enable
R(config-if)# ipv6 address dhcp
(checking)
R(config-if)# end
```

```
R# show ipv6 interface g0/1
...
```

Target. Configure DHCPv6 client to test server.

Note.

DHCPv6 configure router as relay agent

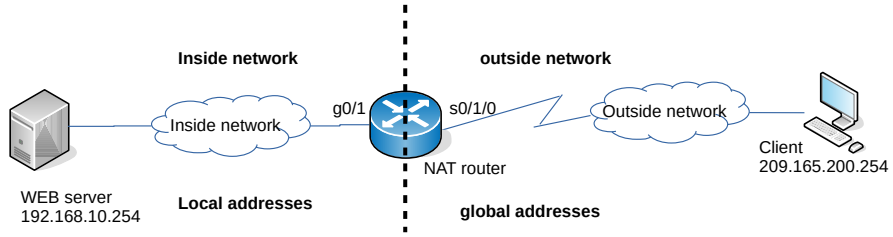
```
R(config)# interface g0/0
R(config-if)# ipv6 dhcp relay destination 2001:db8:cafe:1::6
(checking)
R(config-if)# end
R# show ipv6 interface g0/0
...
    Rely destinations
      2001:DB8:CAFE:1::6
```

Target. Same topology as DHCPv4 relay agent

Note.

Router: NAT

Using this terminology (see ref.4):



Picture 2: private internets terminology

IPv4 configure static NAT

```
R(config)# ip nat inside source static 192.168.10.254 209.165.201.5
R(config)# interface g0/1
R(config-if)# ip address 192.168.10.10 255.255.255.0
R(config-if)# ip nat inside
R(config-if)# interface s0/1/0
R(config-if)# ip address 209.165.200.1 255.255.255.252
R(config-if)# ip nat outside
(checking)
R(config-if)# end
R# show ip nat translations
Pro  Inside global     Inside local       Outside local      Outside global
---  209.165.201.5     192.168.10.254     ---                ---
R# show ip nat statistics
...
```

Target. Configure an inside local address static translated to an inside global address.

Note.

- 1st line: the source inside local address 192.168.10.254 will be translated into the source inside global address 209.165.201.5
- then set interfaces: the inside (lines 2 and 3) and the outside (lines 4 and 5)

IPv4 configure dynamic NAT

```
R(config)# access-list 1 172.16.0.0 0.0.255.255
R(config)# ip nat pool NPN 209.165.76.196 209.165.76.199 netmask 255.255.255.252
R(config)# ip nat inside source list 1 pool NPN
R(config)# interface g0/1
```

```
R(config-if)# ip address 172.16.10.1 255.255.255.0
R(config-if)# ip nat inside
R(config-if)# interface s0/0/0
R(config-if)# ip address 209.165.200.1 255.255.255.252
R(config-if)# ip nat outside
(checking)
R(config-if)# end
R# show ip nat translations
...
```

Target. Configure a dynamic NAT: all 172.16.0.0/16 addresses can be translated through 209.165.76.196/30 addresses (2 IPv4 addresses: ...197 and ...198).

Note.

- 1st line define the permitted addresses using ACL 1;
- 2nd line define the dynamic NAT pool (name, low and high addresses, netmask);
- 3rd line bind the ACL with the pool;
- then, as usual, the inside and outside interfaces.

IPv4 configure port address translation (PAT)

Known also as overload translation.

```
R(config)# access-list 1 172.16.0.0 0.0.255.255
R(config)# ip nat pool NPN 209.165.76.196 209.165.76.199 netmask 255.255.255.252
R(config)# ip nat inside source list 1 pool NPN overload
R(config)# interface g0/1
R(config-if)# ip address 172.16.10.1 255.255.255.0
R(config-if)# ip nat inside
R(config-if)# interface s0/0/0
R(config-if)# ip address 209.165.200.1 255.255.255.252
R(config-if)# ip nat outside
(checking)
R(config-if)# end
R# show ip nat translations
...
```

Target. Configure a port address translation: all 172.16.0.0/16 addresses can be translated through 209.165.76.196/30 addresses (2 IPv4 addresses: ...197 and ...198).

Note. This is similar to IPv4 configure dynamic NAT, key is the keyword **overload** in 3rd line; so we have:

- 1st line define the permitted addresses using ACL 1;
- 2nd line define the dynamic NAT pool (name, low and high addresses, netmask);
- 3rd line bind the ACL with the pool;
- then, as usual, the inside and outside interfaces.

IPv4 configure PAT, single address

```
R(config)# access-list 1 172.16.0.0 0.0.255.255
R(config)# ip nat inside source list 1 interface s0/0/0 overload
R(config)# interface g0/1
R(config-if)# ip address 172.16.10.1 255.255.255.0
R(config-if)# ip nat inside
R(config-if)# interface s0/0/0
R(config-if)# ip address 209.165.200.1 255.255.255.240
R(config-if)# ip nat outside
(checking)
R(config-if)# end
R# show ip nat translations
...
```

Target. Configure a port address translation: all 172.16.0.0/16 addresses can be translated through the single address 209.165.200.1 of the outside interface.

Note. In this case bind is between ACL and the outside interface, not a NAT pool; again, beware of the keyword **overload** in 3rd line; so we have:

- 1st line define the permitted addresses using ACL 1;
- 2nd line bind the ACL with the interface;
- then, as usual, the inside and outside interfaces.

Device management

Cisco discovery protocol (CDP)

```
RS(config)# cdp run
RS(config)# interface g0/1
RS(config-if)# cdp enable
(checking)
RS(config-if)# end
RS# show cdp
…
RS# show cdp neighbors
...
RS# show cdp interface
...
```

Target. Using CDP

Note.

- 1st line assure CDP is running, even if it's so by default; use **no** to disable CDP globally
- 3rd line enable CDP on a specific interface; using **no** it's possible to disable it on the specific interface.

Link layer discovery protocol (LLDP)

```
RS(config)# lldp run
RS(config)# interface g0/1
RS(config-if)# lldp transmit
RS(config-if)# lldp receive
(checking)
RS(config-if)# end
RS# show lldp neighbors
...
RS# show lldp neighbors detail
...
```

Target. Using LLDP

Note.

- 1st line assure LLDP is running;
- 3rd ans 4th lines enable CDP on a specific interface; it's necessary enable transmit and receive severaly.

Set date and time

```
RS# clock set 20:36:00 nov 9 2019
RS# configure terminal
RS(config)# ntp server 193.204.114.232
(checking)
RS(config-if)# end
RS# show clock detail
...
RS# show ntp associations
...
RS# Cshow ntp status
...
```

Target. Set date and time manually or by NTP.

Note.

- 1st line set manually time and date;

- 3rd line set NTP client to ntp1.inrim.it NTP server;

Syslog

```
RS(config)# service timestamp log datetime
RS(config)# logging console
RS(config)# logging buffered
(to syslog server)
RS(config)# logging 192.168.10.3
RS(config)# logging trap 4
RS(config)# logging source-interface g0/0
(checking)
RS(config-if)# end
RS# show logging
...
```

Target. Configuring syslog service

Note.

- 1st line enable to show date and time in log messages; clock must be set before to run this command;
- 2nd and 3rd lines set logging to console and RAM;
- from 4th to 6th lines set sending log messages to syslog server 192.168.10.3, with levels from 0 to 4 and with source IP address as g0/0.

File system

```
RS# show file systems
RS# dir
RS# cd nvram:
RS# pwd
(to syslog server)
RS(config)# logging 192.168.10.3
RS(config)# logging trap 4
RS(config)# logging source-interface g0/0
```

Target. Navigating file system

Note.

Backup and restore

```
RS# copy running-config tftp
...
RS# copy tftp running-config
...
RS# copy running-config usbflash0:
...
RS# copy usbflash0:/R1-Config running-config

RS# copy flash0: tftp:
Source filename []? c1900-universalk9-mz-SPA.152-4.M3.bin
...
RS# copy tftp: flash0:
Address or name of remote host []? 2001:DB8:CAFE:100::99
Source filename []? c1900-universalk9-mz-SPA.152-4.M3.bin
...
RS# license save flash0://all-licenses
(checking)
RS# dir usbflash0:/
...
RS# more usbflash0:/R1-Config
...
```

Target. Backup and restore of configuration files

Note.

- 1st line backup running configuration to tfpt server;
- 3rd line restore running configuration from tftp server;

Password recovery

Steps:

- enter in ROMMON mode
  - connect to console,
  - boot up router,
  - make a break sequence (ex.in PuTTY ctrl-break)
- change configuration register to 0x2142 (ignore startup file) and restart

```
rommon 1 > confreg 0x2142
rommon 2 > reset
```

- change startup file:

```
R# copy startup-config running-config ...
R# configure terminal
R(config)# enable secret cisco
R(config)# config-register 0x2102
R(config)# end
R# copy running-config startup-config
R# reload
```

-

Target. Backup and restore of configuration files

Note.

- 1st line backup running configuration to tfpt server;
- 3rd line restore running configuration from tftp server;

Boot source

```
RS(config)# boot system
            flash0://c1900-universalk6-mz.SPA.152-4.M3.bin
RS(config)# exit
RS# copy running-config startup-config
RS# reload
```

Target. Set from where to boot up system

Note.

Licensing process

Steps:

- purchase software, get Product Activation Key (PAK): an 11 digits alpha numeric key;
- use PAK and Unique Device Identifier (UDI) to get the Software Activation License (a file) using program Cisco License Manager (CLM) or the Cisco License Registration Portal;
- install the license file

```
RS# license install flash0:securityk9-CISCO1941-FHH12250057.lic
RS# reload
(evaluation license installation)
RS(config)# license accept end user agreement
RS(config)# license boot module c1900 technology-package datak9
RS(config)# end
RS# reload
```

Target. Install license

Note.

Other notions

powers of 2

$2^1 = 2$	$2^8 = 256$
$2^2 = 4$	$2^9 = 512$
$2^3 = 8$	$2^{10} = 1024$
$2^4 = 16$	$2^{11} = 2048$
$2^5 = 32$	$2^{12} = 4096$
$2^6 = 64$	$2^{13} = 8192$
$2^7 = 128$	$2^{14} = 16384$

Cidr and block

prefix	bin (x,y)		mask	nets (2^x)	blok size (2^y)	hosts (2^y-2)
/24	0000 0000		0	1	256	254
/25	**1**000 0000	+128	128	2	128	126
/26	**11**00 0000	+64	192	4	64	62
/27	**111**0 0000	+32	224	8	32	30
/28	**1111** 0000	+16	240	16	16	14
/29	**1111 1**000	+8	248	32	8	6
/30	**1111 11**00	+4	252	64	4	2
/31	**1111 111**0	+2	254	128	2	0
/32	**1111 1111**	+1	255	256	1	-1

subnetting: steps

	Question	How calculate
1	n. of subnets	2^x (x=n. of masked bits)
2	How many hosts in subnet	2^y - 2 (y=n. of unmasked bits)
3	list subnets	• 256 - s. mask = block size • 0, 0+block size, 0+2*block size, ... s .mask
4	broadcast addr.	next subnet-1; last is 255
5	valid hosts	from subnet+1 to broadcast-1

subnetting: example

Subnetting 192.168.10.0 using /27.
S. mask 255.255.255.224 (1... 1... 1... 11100 000)
N.of masked bits: x=3
N.of unmasked bits y=5
N.of subnets 2^3=8
N.of hosts in subnet 2^5-2=32-2=30
Block size 256-224=32

net	broadcast	1st host	last host

0	31	1	30
32	63	33	62
64	95	65	94
96	127	97	126
128	159	129	158
160	191	161	190
192	223	193	222
224	255	225	254

variable length subnetting

Given the base network, for each subnet do the following:

- how many hosts in the subnet? we determine the block size as hosts+2 and searching the minimum block size that contains it;
- from the block size, we get prefix and s.mask.

Then we list previous values as follow, allocating subnets from the largest to the smallest block, one after the other, without overlapping

net	hosts	block	prefix	mask	base	subnet
A						
B						
...						

Example. 3 subnets: A, 25 hosts, B 5 hosts, C 2 hosts. Base lan: 192.168.10.0

net	hosts	block	prefix	mask		subnet
A	25	26+2=27 → 32	/27	224	0	192.168.10.0/27
B	5	5+2=7 → 8	/29	248	0+32	192.168.10.32/29
C	2	2+2=4	/30	252	32+8	192.168.10.40/30
					40+2	

Interfaces

From rif.1. Convention to identify interface is:

- interface on WIC: type slot_#/subslot_#/port_#
- interface on chassis: type slot_#/port_#

type	shortcut	example and note
serial	s	serial 0/0/0
gigabitethernet	g	gigabitethernet 0/1
fastethernet	fa	fastethernet 0/1
vlan	vlan	vlan 1

Environments

prompt	what is	how enter in it
`config`	global configuration	`R# configure terminal`
`config-if`	configure interface	`R(config)# interface` *`interf-id`*

prompt	what is	how enter in it
`config-if-range`	Configure a range of interfaces	`S(config)# interface range interf/1stport-lastport`
`config-router`	configure routing module	`R(config)# router` *`dynamic-protocol`*
`config-std-nacl`	Configure standard named acl	`R(config)# ip access-list standard acl-name`
`config-vlan`	Configure VLAN	`R(config)# vlan` *`vlan-id`*
`dhcp-config`	Configure DHCP pool	`R(config)# ip dhcp pool pool-name`

notably addresses

IPv4

- Default static route - 0.0.0.0 0.0.0.0
- private internets
  - class A, from 10.0.0.0 to 10.255.255.255: 10.0.0.0/8
  - class B, from 172.16.0.0 to 172.31.255.255: 172.16.0.0/12
  - class C, from 192.168.0.0 to 192.168.255.255: 192.168.0.0/16

IPv6

- Local link - FE80::/8
- multicast routing - FF00::/8
- default static route - ::/0

IPV4 address schema

type	block	from		to		How many addresses		Note
		Decimal	Binary	Decimal	binary	Power of 2	decimal	
unicast	0.0.0.0/3	0.0.0.0	0… 0… 0… 0…	223.255.255.255	1101 1111.1… 1… 1…	2^29	536,870,912	
private	10.0.0.0/8	10.0.0.0		10.255.255.255		2^24	16,777,216	
	172.16.0.0/12	172.16.0.0		172.31.255.255		2^20	1,048,576	
	192.168.0.0/16	192.168.0.0		192.168.255.255		2^16	65,536	
special	127.0.0.0/8	127.0.0.0		127.255.255.255		2^24	16,777,216	loopback
	169.254.0.0/16	169.254.0.0		169.254.255.255		2^16	65,536	Link local
	192.0.2.0/24	192.0.2.0		192.0.2.255		2^8	256	Test networks
multicast	224.0.0.0/24	224.0.0.0	1110 0000.0… 0… 0…	239.255.255.255	1110 1111.1… 1… 1…	2^28	268,435,456	Class D. MAC: [2]
experimental	240.0.0.0/4	240.0.0.0	1111 0000.0… 0… 0…	255.255.255.254	1… 1… 1… 1111 1110	2^28-1	268,435,455	Class E
broadcast	255.255.255.255/32	255.255.255.255		255.255.255.255		2^0	1	MAC: [3]

Unicast addresses include private and special addresses. So multicast starts from where unicast ends.

2 `01.00.5E`+(23 least significant bits of IP)
3 `FF.FF.FF.FF.FF.FF`

notably ports

- 0-511, 1sr half well known ports;
- 512 1023, 2nd half well known ports
- 1024-65535, registered ports (to 49151)+dynamic or private ports

Incoherences

- asking about what dynamic routing protocols router supports: IPv4 has a different syntax compared to IPv6:
  - `R(config)# router ?`
  - `R(config)# ipv6 router ?`
- Defining ACLs in routers, there are two different kind of definitions for numbered and named ACLs:
  - numbered: `R(config)# access-list 10 permit 192.168.10.0 0.0.0.225`
  - named:
    - `R(config)# ip access-list standard EXAMPLE`
    - `R(config-std-nacl)# permit 192.168.10.0 0.0.0.225`

Observations

- Commands setting something, usually accept the form `no` *`command`*. This form negate the command destroying its result.
- Primary IPv4 route learning modes, with Administrative Distances in parenthesis:
  - **C** - active directly connected interface (0);
  - **L** - locale: active interface with IPv4 address ;
  - **S** - static, manually configured (1);
  - **EX** - dynamic, EIGRP external (20);
  - **B** - dynamic,learnt by BGP (20);
  - **D** - dynamic, learnt by EIGRP (90);
  - **D** - dynamic, learnt by IGRP (100);
  - **O** - dynamic, learnt by OSPF (110);
  - **i** - dynamic, IS_IS (115);
  - **R** - dynamic, learnt by RIP (120);
  - ***** - candidate default.
- `::/0` - network match for a default static IPv6 route

Commands

List

List of used commands, ordered alphabetically first by context (without the system type: R or S or both), then by command.

context	command	note
R#	**clear access-list counters**	
R#	**debug ipv6 dhcp detail**	
S#	**delete flash:vlan.dat**	
S#	**erase startup-config**	
RS#	**show ... | {section | include | exclude | begin}** *a_filter*	Filter show output based on the filter on right side of the pipe sign
R#	**show access-list** [*acl-id*|*acl-name*]	
S#	**show controllers ethernet-controller** *interf-id* **phy**	doesn't function in PT
RS#	**show ip interface** *interf-id*	
R#	**show ip route**	shows router table. accepts ipv6
RS#	**show ip ssh**	
R#	**show ipv6 dhcp pool**	
R#	**show ipv6 interface** *interf-id*	
S#	**show port-security address**	show what MAC addresses are associated with ports
S#	**show port-security interface** *interf-id*	
RS#	**show vlan** [**brief**]	On Packet Tracer router this command doesn't show vlan on subinterfaces
RS(...)#	**end**	to the base environment
RS(...)#	**exit**	to the previous environment
S(config)#	**access-list** *acl-id* {**remark**|**permit**|**deny**} [**host**|**any**] *source* [*wildcard*] [**log**]	*source* is the network from which the packet is sent. *wildcard* is incompatible with keywords **host|any**. host is wildcard 0.0.0.0, any is wildcard 255.255.255.255
RS(config)#	**boot system** *drive:/path-to-IOS-file/IOS-file*	Set BOOT environment variable
RS(config)#	**crypto key generate rsa general-keys modulus** *length*	Generate private/public keys pair. It needs `ip domain-name ...`
RS(config)#	**do** *command*	Execute a command from global configuration mode
RS(config)#	**enable** {**secret**|**password**} *pwd*	Set {clear or crypted} password for privileged mode (enable: you see)
RS(config)#	**hostname** *name*	Set hostname of system
RS(config)#	**interface** *interf-id*	go to interface configuration. *interf-id* is interface id (see chapter: Interfaces)

context	command	note
RS(config)#	**ip default-gateway** *addr*	set default gateway. In case of router only if routing is disabled
RS(config)#	**ip domain-name** *name.ext*	Set domain name of the device (used by `crypto key generate ...`)
RS(config)#	[**no**] **ip domain-lookup**	Disable DNS lookup
RS(config)#	**ip domain-name** *name.com*	Set domain of the system
R(config)#	**ip route** *network nmask* {*next-hop-ip* | *exit-interface*} [*AD*] [**permanent**]	set a IPv4 static route. AD is administrative distance. permanent: reload this route on bootstrap
RS(config)#	**ip ssh version 2**	
R(config)#	**ipv6 route** *ipv6-net/len* {*ipv6-next-hop* | *exit-interface*} [*AD*] [**permanent**]	set a IPv6 static route. AD is administrative distance. permanent: reload this route on bootstrap
R(config)#	**ipv6 unicast-routing**	enable IPv6 routing
RS(config)#	**line** {**console**|**vty** ***1st-num last-num***|**aux** ***num***}	
RS(config)#	**login block-for** *block-secs* **attempts** *n* **within** *secs*	
R(config)#	**router ?**	show supported dynamic routing algorithms. use `ipv6 router ?` in case of IPv6
R(config)#	**router rip**	enter in router configuration, RIPv1 enabled
RS(config)#	**security password min-length** *len*	
RS(config)#	**service password-encryption**	
RS(config)#	[**no**] **username** *name* **privilege** *level* **secret** *password*	Level max 15
S(config)#	[**no**] **vlan** *vlan-id*	Create a new VLAN, then enter in vlan configuration mode. The **no** argument destroy it (before move its ports to another VLAN)
S(config-if)#	**duplex** [**auto**|**full**|**half**]	
RS(config-if)#	**ip address** *addr nmask*	set IPv4 network address on interface
S(config-if)#	**ip access-group** {*acl-id*|*acl-name*} {**in**|**out**}	
R(config-if)#	**ip helper-address** *ipv4-addr*	
RS(config-if)#	**ipv6 address** *ipv6addr/len* [**link-local**]	set IPv6 network address. If link-local parameter is present, it sets a link-local address
RS(config-if)#	**ipv6 enable**	enable IPv6 address management on the interface. It assign a link-local address automatically generated
S(config-if)#	**mdix** [**auto**|...]	
S(config-if)#	**mls qos trust cos**	enable Quality of Service (qos) driven by the Class of Service (cos) of the Voip phone
RS(config-if)#	[**no**] **shutdown**	turns on the interface. shutdown turns off
S(config-if)#	**speed** [**auto**|**10**|**100**|...]	
S(config-if)#	**switchport access vlan** *vlan-id*	Set port membership to indicated VLAN id. If port had a different membership, it changes
S(config-if)#	**no switchport access vlan**	Set port membership to VLAN1

context	command	note
S(config-if)#	**switchport mode** [**access** | **trunk** |...]	
S(config-if)#	**switchport port-security mac-address** [**sticky**] [*MAC*]	
S(config-if)#	**switchport port-security violation** {**protect**|**restrict**|**shutdow**}	
S(config-if)#	[**no**] **switchport trunk allowed vlan** *vlan-list*	Key argument **no** and *vlan-list* are mutually exclusive
S(config-if)#	[**no**] **switchport trunk native vlan** *vlan-id*	Key argument **no** and *vlan-id* are mutually exclusive
S(config-if)#	**switchport voice vlan** *vlan-id*	configure automatically the port like a mini trunk, with two VLANs: the voice and the data (see ref.3)
R(config-line)#	**access-class** *acl-num* {**in**|**out**}	Use ACL *acl-num* to restrict access to the current line. Note: ACLs are at layer 3, so do not try to use them on switch at layer 2
R(config-line)#	**clock rate** *clock-rate*	Use on DCE side of asynchronous line. Rate example: 128000
RS(config-line)#	**logging synchronous**	To prevent console messages from interrupting commands
RS(config-line)#	**login** [**local**]	Use password to login. If **local**, use locally configured user and his password
RS(config-line)#	**password** *pwd*	Assign *pwd* password to use the current line
RS(config-line)#	**transport input ssh**	
R(config-router)#	**default-information originate**	propagate the default static address (gateway of last resort)
R(config-router)#	**network** *a-network*	present network to RIP, hook the relative interface. *a-network* is in the form 192.168.15.0. RIPv2 admits classless networks
R(config-router)#	**no auto-summary**	disable auto-summary mode (RIPv2 only)
R(config-router)#	**passive-interface** *interf-id*	do not propagate RIP messages using this interface
R(config-router)#	**version 2**	set RIPv2
S(config-vlan)	**name** *vlan-name*	

Synthesis

Routing	IPv4	static	ip route *net-addr next* permanent		
		dynamic	router rip * version 2 *+ network *net-addr* * passive-interface * default-information originate		
	IPv6	static	ipv6 route *net-addr next* permanent		
Switch Port security		secure port static	interface interf-id * switchport mode access * switchport port-security * switchport port-security mac-address *mac* * switchport port-security violation *protect*		
		secure port sticky	interface interf-id * switchport mode access * switchport port-security * switchport port-security mac-address sticky * switchport port-security violation *protect*		
Switch VLAN			vlan *vlan-id* * name *vlan-name*	port assign	interface *interf-id* * switchport mode access * switchport access vlan *vlan-id*
				trunk assign	interface *interf-id* * switchport mode trunk * switchport trunk native vlan *vlan-id* * switchport trunk allowed vlan *vlan-list*
Router ACL	IPv4	numeric	access-list *num-id* permit *source wildcard* access-list *num-id* deny *source wildcard* …	interface assign	interface *interf-id* * ip access-group *id* {in | out}
		named	ip access-list standard acl-name * permit source wildcard * deny source wildcard		

Router DHCP	IPv4	server	ip dhcp excluded-address *low-addr high-addr* … ip dhcp pool *pool-name* * network *net-addr* * default-router *addr* * dns-server *addr* * domain-name *name* * lease *time* * netbios-name-server *addr* service dhcp	relay agent	interface interf-id * ip helper-address addr
				client	interface interf-id * ip address dhcp
	IPv6	stateless server	ipv6 unicast-routing Ipv6 dhcp pool *pool-name* * dns-server *addr* * domain-name *name*	interface assign	interface interf-id * ipv6 enable * ipv6 addr * ipv6 dhcp server pool-name * ipv6 nd other-config-flags
				client	interface interf-id * ipv6 enable * ipv6 address autoconfig
		stateful server	ipv6 unicast-routing Ipv6 dhcp pool *pool-name* * address prefix *addr-prefix/len* * dns-server *addr* * domain-name *name*	interface assign	interface interf-id * ipv6 enable * ipv6 addr * ipv6 dhcp server pool-name * ipv6 nd managed-config-flags
				client	interface interf-id * ipv6 enable * ipv6 address dhcp
				relay agent	interface interf-id * ipv6 dhcp relay destination addr
Router NAT	IPv4	static	ip nat inside source static *local-ip global-ip* …	interface assign	interface interf-id * ip nat inside Interface interf-id * ip nat outside
		dynamic	ip nat pool *pool-name low-addr high-addr* {netmask *nmask*|prefix-length *len*} access-list *acl-num* permit *source* [*wildcard*] ip nat inside source list *acl-num* pool *pool-name*		
		PAT	ip nat pool *name low-addr high-addr* {netmask *nmask* | prefix-length *len*} access-list *acl-num* permit *source* [*wildcard*] ip nat inside source list *acl-num* pool *name* overload		
		PAT sing addr	access-list *acl-num* permit *source* [*wildcard*] ip nat inside source list *acl-num* interface *interf-id2* overload	interface assign	interface interf-id * ip nat inside Interface interf-id2 * ip nat outside
		port forwarding	ip nat inside source static {tcp|udp} local-ip local-port global-ip global-port	interface assign	interface interf-id * ip nat inside Interface interf-id2 * ip nat outside

References

Bibliography

ref.	Title	Author	Note
1	Interface Nomenclature Guide of Router and Switch	Computer networking notes	https://www.computernetworkingnotes.com/ccna-study-guide/interface-nomenclature-guide-of-router-and-switch.html @ https://www.computernetworkingnotes.com/
2	Attribution-ShareAlike 4.0 International	creative commons	this document license: CC BY-SA 4.0 https://creativecommons.org/licenses/by-sa/4.0/ @ https://creativecommons.org/
3	Switchport Voice VLAN – What Does It Do?	Tom Hollingswoth	@ https://networkingnerd.net/2012/05/09/switchport-voice-vlan-post/
4	RFC1918 - Address Allocation for Private Internets	IETF	https://tools.ietf.org/html/rfc1918
5	RFC 2131 – Dynamic Host Configuration Protocol	IETF	https://www.rfc-editor.org/rfc/rfc2131.txt

Acronyms and abbreviations

Acronyms	Means
AD	Administrative Distance
addr	address IPv4 (or IPv6)
CoS	Class of Service
IETF	Internet Engineering Task Force
interf-id	interface identifier
IOS	Cisco Internetwork Operating system
nmask	networking mask IPv4
NTP	Network Time Protocol
PAK	Product Activation Key
PID	Product ID
PT	Packet Tracer
QoS	Quality of Service
SN	Serial Number
Udi	Unique Device Identifier: Product ID+Serial Number+hardware version
VLAN	Virtual LAN