Cheatsheet Packet Tracer/Cisco

RL
crunch it connected

Mode Navigation

R> `enable`
//enters the Privileged EXEC mode
R# `configure terminal`
//enters the global config mode
R(config)# `interface <type>/<number>`
//enters the interface type/number config mode

Example: `interface fa0/1`

Tips and Tricks

`?`
//displays all the possible commands in the current mode
`<tab>`
//autocompletes the rest of the command
`do <command>`
//executes the command in the Privileged Exec mode, regardless of the current shell mode
`<shortcut>`
//you can execute a command by typing just the first letters of it and press enter
`<click Fast Forward Time>`
//increases the time of booting the devices
`exit`
//exits the current mode
`end`
//exits the current mode and enters the Privileged EXEC mode
`<CTRL+SHIFT+6>`
//interrupts the execution of the current command
`no <command>`
//cancels the command/ deletes the configuration of that command

Example: `en`
`conf t`
`int fa0/0`

Show commands

`show running-config`
//view the router's/switch's entire active configuration

`show ip interface brief`
//view the available interfaces and their brief parameters (IP, active, etc.)

`show ip route`
//view the routing table

`show mac-address-table`
//view the CAM table

`show spanning-tree`
//view spanning-tree (STP) parameters

`show VLAN brief`
//view VLAN parameters

`show interface VLAN brief`
//view VLAN's brief parameters on interfaces

Basic commands

#ADD IPs (on router's interfaces)

R(config)# `interface <type>/<number>`
//enters the interface config mode
R(config-if)# `ip address <IP> <decimal-MASK>`
//sets the IP and the mask to the interface
R(config-if)# `no shutdown`
//enables the interfaces (brings it up)

Example: `int fa0/3`
`ip add 10.10.10.1 255.255.255.248`
`no shut`

Spanning Tree Protocol

`spanning-tree vlan <vlan-number> priority <value>`
//sets the priority <value> of the switch for the STP by vlan

VLAN Configuration
(only on Switch)

#MODE ACCESS (interfaces connected to end-devices)

`vlan <vlan-number>`
//creates the VLAN
`interface <type>/<number>`
//enters the interface that needs to be configured
`switchport mode access`
//sets the access mode
`switchport access vlan <vlan-number>`
//sets the access vlan

Example: `vlan 10`
`int fa0/2`
`sw mo acc`
`sw acc vlan 10`

#MODE TRUNK (interfaces connected to other switches or routers)

`interface <type>/<number>`
enters the interface that needs to be configured
`switchport mode trunk`
sets the trunk mode
`swithcport trunk allowed vlan <vlan-number>/all`
sets the vlans that are allowed on that link (some vlans or a range or vlans or all vlans)

Example: `vlan 10`
`int fa0/1`
`sw mo tr`
`sw tr allowed vlan 10`
or
`sw tr allowed vlan all`
or
`sw tr allowed vlan range 10-20`

#MANAGEMENT VLAN (configuration on switch)

`interface vlan <vlan-number>`
//enters the VLAN interface
`ip address <IP> <MASK>`
//assigns the IP address and mask
PC> `telnet <IP>`
//connects to the switch's IP

Example: `int vlan 99`
`ip add 10.10.10.99 255.255.255.0`

PC> `telnet 10.10.10.99`

CAM Table

`mac-address-table static <MAC address> vlan <vlan-number> interface <type>/<number>`
//the MAC address will be stored as static in the CAM table
`clear mac-address-table`
//flush the contents of the CAM table

Example: `en`
`conf t`
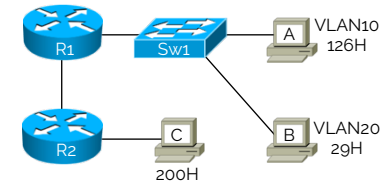`mac-address-tabel static 0001.6458.8b1a vlan 10 int fa0/1`

Routing Configuration

`ip route <destination network> <destination network's mask> <next-hop>`
//sets the route to the destination network through the next-hop

Example: `en`
`conf t`
`ip route 10.10.10.0 255.255.255.0 192.168.0.1`

`ip route 0.0.0.0 0.0.0.0 <next-hop>`
//sets the default route: all the packets with unknown destinations will be sent through that next-hop

Subnetting 101

192.168.0.0/22
R1R2 – one network – 2H
VLAN10 – one network – 126H
VLAN20 – one network – 29H
R2C – one network – 200H

- add default gateway and extra 2
- write them in descending order
- find closest power of 2
  - $200+1+2 <= 2^8$
  - $126+1+2 <= 2^8$
  - $29+1+2 <= 2^5$
  - $2+2 <= 2^2$
- the power of 2 represents the mask
  - 32-power -> /mask

Example:
  - power is 6
  - then mask is /26

- R2C
  - 192.168.0.0/24 -> 192.168.0.255/24
- VLAN10
  - 192.168.1.0/24 -> 192.168.1.255/24
- VLAN20
  - 192.168.2.0/27 -> 192.168.2.31/27
- R1R2
  - 192.168.2.32/30 -> 192.168.2.35/30

Cheatsheet Linux - Networking

VM Setup

http://bit.ly/openstack_rl_tutorial

`ssh -o ServerAliveInterval=100 <ldap_user>@fep.grid.pub.ro`
//connect to your fep account
`ssh -i ~/.ssh/openstack.key student@<IP_masina_virtuala>`
//connect to the virtual machine you just created in Openstack

Example: `ssh -o ServerAliveInterval=100 adi.minune@fep.grid.pub.ro`

`ssh -i ~/.ssh/openstack.key student@10.9.24.226`

Tips and Tricks

`go [red|green|blue]`
//connect to one of the 3 containers
`lxc-list`
//view the list of containers and their state
`rr [red|green|blue]`
//reboot one of the 3 containers
`<shortcut>`
//you can execute a command by typing just the first letters of it and press enter
`<CTRL+a> -> <press q>`
//exit the console of the container
`ping -c <value> <IP>`
//test the conectivity between host and <IP> by sending <value> packets

Example: `ping -c 2 10.10.0.1`

Basic commands

#ADD IPs

`ip address add <IP>/<MASK> dev <interface>`
//sets the IP and the mask to the interface
`ip address flush dev <interface>`
//resets the interface at the initial configuration
`ip link set dev <interface> up`
//enables the interface
`ip route add default via <IP-default-gateway>`
//sets the default gateway
`sysctl -w net.ipv4.ip_forward=1`
//activates routing/packet forwarding

Example: `ip add add 192.168.0.1/24 dev veth-red`
`ip l s dev veth-red up`
`ip r a default via 10.0.0.1`

Show commands

`ip address show dev <interface>`
//view the layer 3 (network) configuration of the interface

`ip link show dev <interface>`
//view the layer 2 (data link) configuration of the interface

`ip route show`
//view the routing table

`ip neighbor show`
//view the ARP table

Example: `ip a s dev eth0`
`ip l s dev veth-red`
`ip r s`

Network Services

#REMOTE CONNECTION

`ssh <username>@<IP/hostname> -p <port-number>`
//connects to <username> at remote <IP/hostname> via ssh on port <port-number>
`ssh -l <username> <IP/hostname>`
//connects to <username> at remote <IP/hostname> via ssh
`ssh-keygen -t rsa`
//generates public/private rsa key pair
`ssh-copy-id <username>@<IP/hostname>`
//copy public key in the remote file for authentication on <username> at <IP/hostname>
`telnet <IP/hostname>`
//connects to <IP/hostname> via telnet
`ftp <IP/hostname>`
//connects to <IP/hostname> via ftp
`scp -r <username>@<hostname>:<folder>`
//downloads <file> from <username> at <hostname> on your local host
`scp -r <file> <username>@<hostname>:`
//uploads <file> from local host to <username> at <hostname>

#TRAFFIC CAPTURE

`netcat`
//arbitrary TCP and UDP connections and listens
`-l`   listens (server) to connections
`-u`   use UDP instead of the default option of TCP
`netstat`
//prints network connections
`-t`   lists TCP connections
`-l`   lists services that listen on connections
`-u`   lists UDP connections

`dsniff -I <interface>`
//captures network traffic and lists credentials when connections end

Example:
`netcat -l 1234`
//server that listens on TCP port 1234
`netstat -tlnp`
//lists the TCP services that listen on port 2024
`dsniff -I eth0`

IPTABLES

`iptables -t [table] [-A|-D|-I|-R|-L|-F] [chain] [options] [action]`

`-t filter`   filtering table (the default table)
`-t nat`   altering table
`-t mangle`   special altering table

`-A`   append rule to chain
`-D`   delete rule
`-I <no>`   insert as the given rule <no>
`-R`   replace rule
`-L`   list all rules from given chain
`-F`   flush the selected chain

#filter chains:
`INPUT`   packets destined to local host
`OUTPUT`   packets locally-generated
`FORWARD`   packets being routed through the local host

#nat chains:
`PREROUTING`   altering packets as soon as they come in
`POSTROUTING`   altering packets as they are about to go out
`OUTPUT`   altering locally-generated packets before routing

`-d <IP/hostname>`   destination option
`-s <IP/hostname>`   source option
`-p [tcp|udp|icmp|all|<number>]`   protocol option
`-i <input-interface>`   input interface option
`-o <output-interface>`   output interface option
`--dport <protocol/number>`   destination port
`--sport <protocol/number>`   source port

#actions:
`-j ACCEPT`   let through the packet that matched the options
`-j REJECT`   rejects the packet that matched the options
`-j DROP`   drops the packet, without sending a notification error
`-j DNAT`   available only in the nat table, it specifies that the destination address of the packet should be modified

Example:
`iptables -L FORWARD -n -v`
//view (list) rules and information on the filter table – FORWARD chain

`iptables -A INPUT -p tcp -d 10.10.0.1 –dport 21 -s 20.20.0.1 -j DROP`
//add a rule to block FTP (port 21) from 20.20.0.1 to 10.10.0.1

`iptables -t nat -A PREROUTING -p tcp –dport 22022 -j DNAT –to-destination 10.0.0.1:22`
//add a rule where connections to port 22022 will be redirected to ssh (port 22) of 10.0.0.1