**Cisco IOS Commands**

1. When you start the console, will see the prompt of the User EXEC mode (Router>). To see which commands are available in this mode, type “?”:

**Router>** ?

1. To view and change system parameters of a Cisco router, you must enter the **privileged EXEC mode**, by typing:

**Router>** enable  
**Router#**

1. To modify system wide configuration parameters, you must enter the **global configuration mode**. This mode is entered by typing:

**Router#** configure terminal

**Router(config)#**

1. To make changes to a network interface, enter the **interface configuration mode**, with the command:

**Router(config)#** interface FastEthernet0/0  
**Router(config-if)#**

The name of the interface is provided as an argument. Here, the network interface that is configured is FastEthernet0/0.

1. To **return** from the interface configuration to the global configuration mode, or from the global configuration mode to the privileged EXEC mode, use the “exit” command:

**Router(config-if)#** exit  
**Router(config)#** exit  
**Router#**

Or, to directly **return** to the privileged EXEC mode from any configuration mode, use the “end” command:

**Router(config-if)#** end  
**Router#**

1. To **return** from the privileged EXEC mode to the user EXEC mode, type:

**Router#** disable  
**Router>**

1. To **terminate** the console session from the user EXEC mode, type:

**Router>** logout

Router con0 is now available  
Press RETURN to get started.

Or type “logout” or “exit” from the privileged EXEC mode:

**Router#** exit

Router con0 is now available  
Press RETURN to get started.

**IOS MODE: PRIVILEGED EXEC Mode (Router#)** you can check your **router configuration** using the following commands:

**show interfaces  
show running-config**

**show ip route**

Displays the contents of the routing table

**clear ip route \***

Deletes all routing table entries

**show ip cache**

Displays the routing cache

**clear ip cache**

Clears the routing cache

**show arp**

Displays the arp cache

**clear arp-cache**

Deletes the entire ARP cache, but sends multiple gratuitous arp messages to deleted entries that are still in the IP route cache to verify that they are still alive – so ARP cache is populated again. To stop that from happening:

* Clear the IP route cache: **clear ip cache**, before clearing the arp cache: **clear arp-cache**
* One can also set a very short timoute value for cached arp entries that will delete them within seconds. See commands under Interface configuration.
* We can also disable IP route caching. That way “**clear arp-cache**” command will clear entries and not send out gratuitous arps.

\*\*

**Not used: clear ip arp**

Deletes the entire ARP cache, but sends ONE gratuitous arp to deleted entries

**Not used: clear ip arp force-delete**

Deletes the entire ARP cache with no refresh (i.e., no gratuitous arp for deleted entries)

\*\*

Pinging from a Router interface:

**Router#** ping **IP\_addr** repeat **num**

Without “repeat **num**”, it will send 5 pings. To abort ping:Type Ctrl + Shift + 6

**IOS MODE: GLOBAL CONFIGURATION (Router(config)# )**

**ip route-cache**

Enables route caching. By **default, route caching is enabled** on a router.

**no ip route-cache**

Disables route caching

arp **IPaddress H/WAddress** arpa

no arp **IPaddress H/WAddress** arpa

Adds or deletes a **static** ARP entry for IPaddress with MAC in the ARP cache. These entries need to be removed individually if ARP cache is to be cleared.

Adding a static host or network route via a next hop gateway (router):

ip route **IPAddress netmask gw\_address**

ip route **NetAddress netmask gw\_address**

Adding a static host or network route via a connected interface:

ip route **IPAddress netmask Iface**

ip route **NetAddress netmask Iface**

Adding a default route via a next hop gateway (router):

ip route 0.0.0.0 0.0.0.0 **gw\_address**

Deleting static route entries:

Commands same as above but use "no ip route" instead of "ip route"

**IOS MODE: INTERFACE Configuration (Router(config-if)# )**

Setting arp timeout duration in arp cache

**Router(config-if)#** arp timeout **secs**

Specifies, in units of seconds, how long dynamic entries learned on an interface remain in the Address Resolution Protocol (ARP) cache. For e.g., **secs** = 10. To remove the arp timeout and restore default values, use the **no** form of this command.

Ignoring gratuitous arps

**Router(config-if)#** arp gratuitous ignore

To ignore the receipt of gratuitous Address Resolution Protocol (ARP) packets. To receive gratuitous ARP packets, use the **no** form of this command.

Taking an interface down and bringing it back up:

**Router(config-if)#** shutdown

**Router(config-if)#** no shutdown

Enabling and disabling proxy arp (enabled by default on every interface):

**Router(config-if)#** ip proxy-arp

**Router(config-if)#** no ip proxy-arp

Configuring a Router’s IP addresses on an interface:

**Router>** enable  
**Router#** configure terminal (**OR** shorthand -> conf t)

**Router(config)#** no cdp run

**Router(config)#** no ip routing  
**Router(config)#** ip routing  
**Router(config)#** interface **Iface** (**OR** shorthand -> int **Iface**)  
**Router(config-if)#** ip address **IPAddress** **netmask**

**Router(config-if)#** arp timeout **secs**  
**Router(config-if)#** no shutdown  
**Router(config-if)#** end

**Configuring a Router as a Bridge (shown here for 2 interfaces):**

For example, assigning a **priority of 128**, **spanning tree disabled**, running **protocl ieee**. The first command assigns this bridge to bridge group 1

**Router>** enable

**Router#** configure terminal

**Router(config)#** no ip routing

**Router(config)#** bridge **1** protocol **ieee**

**Router(config)#** bridge **1** priority **128**

**Router(config)#** interface **Iface1**

**Router(config-if)#** bridge-group **1**

**Router(config-if)#** bridge-group **1** **spanning-disabled**

**Router(config-if)#** no shutdown

**Router(config-if)#** interface **Iface2**

**Router(config-if)#** bridge-group **1**

**Router(config-if)#** bridge-group **1** **spanning-disabled**

**Router(config-if)#** no shutdown

**Router(config-if)#** end

**Router#** clear bridge **1**

**Router#** clear arp-cache

**Now we enable spanning tree:**

**Router#** configure terminal

**Router(config)#** interface **Iface1**

**Router(config-if)#** bridge-group 1

**Router(config-if)#** no bridge-group 1 spanning-disabled

**Router(config-if)#** interface **Iface2**

**Router(config-if)#** bridge-group 1

**Router(config-if)#** no bridge-group 1 spanning-disabled

**Router(config-if)#** end

**Router#**

**Displaying bridge configuration and tables**:

**show bridge**

Displays the entries of the MAC forwarding table

**show spanning-tree**

**show spanning-tree brief**

Displays the spanning tree topology information as seen by this bridge

**show interfaces**

Displays statistics of all interfaces, including the MAC addresses of all interfaces

**Routing Commands (Router#)  - OSPF**

**show ip ospf**

Displays general information about the OSPF configuration

**show ip ospf database**

Displays the link state database.

**show ip ospf border-routers**

Displays the Area Border Router (ABR) and Autonomous System Boundary Router (ASBR).

**clear ip ospf process-id process**

Resets the specified OSPF process.

**Enabling and disabling OSPF configuration (Router(config)#)**

router ospf **process-id**

no router ospf **process-id**

**Configuring a Router to use OSPF for dynamic routing**

**Router>** enable

**Router#** configure terminal

**Router(config)#** no ip routing

**Router(config)#** ip routing

**Router(config)#** router ospf 1

**Router(config-router)#** network **Netaddr** **InvNetsask** area **AreaID**

**Router(config-router)#** interface **Iface1**

**Router(config-if)#** no shutdown

**Router(config-if)#** ip address **IPAddress** **netmask**

**Router(config-router)#** interface **Iface2**

**Router(config-if)#** no shutdown

**Router(config-if)#** ip address **IPAddress** **netmask**

**Router(config-if)#** end

**Router#** clear ip route \*

**Routing Commands  - BGP:**

**Router#** show ip bgp

**Router#** show ip bgp summary

**Router#** clear ip bgp \*

**Router#** show ip bgp neighbors

**Router#** show ip bgp paths

**Router(config-router)#** timers bgp **keepalive holdtime**

**Setting up a Router to run BGP:**

**Router>** enable

**Router#** configure terminal

**Router(config)#** no ip routing

**Router(config)#** ip routing

**Router(config)#** interface **Iface1**

**Router(config-if)#** no shutdown

**Router(config-if)#** ip address **IPAddr** **netmask**

**Router(config-if)#** interface **Iface2**

**Router(config-if)#** no shutdown

**Router(config-if)#** ip address **IPaddr** **netmask**

**Router(config-if)#** router bgp **ASnumber**

**Router(config-router)#** neighbor **IPaddrNeighbor** remote-as **ASnumberNeighbor**

**Router(config-router)#** neighbor **IPaddrNeighbor** remote-as **ASnumberNeighbor**

**Router(config-router)#** network **Netaddr** mask **netmask**

**Router(config-router)#** end

**Router#** clear ip bgp \*

**Router#** clear ip route \*

**Linux Commands**

ARP Cache Related:

**arp -a**

Display the content of the ARP cache.

**arp –d IPAddress**

Deletes the entry with the IP address **IPAddress**.

**arp –s** **IPAddress MACAddress**

Adds a static entry to the ARP cache that is never overwritten by network events. The MAC address is entered as 6 hexadecimal bytes separated by colons.

**Example:** arp –s 10.0.1.12 00:02:2D:0D:68:C1

**ip –s –s neigh flush all**

Clears the arp cache

Checking on the network status of the PC:

**netstat –i**

Displays a table with statistics of the currently configured network interfaces.

**netstat –rn**

Displays the kernel routing table. The *–n* option forces netstat to print the IP addresses. Without this option, netstat attempts to display the host names.

**netstat –an  
netstat –tan**

**netstat -uan**

Displays the active network connections. The *–a* option display all active network connections, the *–ta* option displays only information on TCP connections, and the *–tu* option displays only information on UDP traffic. Omitting the *–n* option prints host names, instead of IP addresses.

**netstat –s**

Displays summary statistics for each protocol that is currently running on the host.

Configuring a PC interface:

**ifconfig**

Displays the configuration parameters of all active interfaces.

**ifconfig** **interface**

Displays the configuration parameters of a single interface. For example, ifconfig **eth0** displays information on interface **eth0**.

**ifconfig** **interface** **down**

Disables the interface. For example: ifconfig **eth0** down. No traffic is sent or received on a disabled interface.

**ifconfig** **interface** **up**

Enables an interface.

**ifconfig** **interface** **IPAddress/xx**

**e.g.** ifconfig **eth0 10.0.1.8/24**

Assigns interface *eth0* the IP address *10.0.1.8/24* and a broadcast address of *10.0.1.255*

**ifconfig** **interface** **mtu** **xxx**

Assigns MTU size **xxx** bytes to **interface**

Enabling and Disabling IP Forwarding on a Linux PC:

sysctl net.ipv4.ip\_forward

shows current status of ipforwarding

echo 1 > ‘/proc/sys/net/ipv4/ip\_forward’

enables IPforwarding

echo 0 > ‘/proc/sys/net/ipv4/ip\_forward’

disables IPforwarding

Additional commands that do the same as the above commands:

sysctl -w net.ipv4.ip\_forward=1

sysctl -w net.ipv4.ip\_forward=0

Setting Path Discovery (MTU probing) on a Linux PC:

**PC% echo 2 > `/proc/sys/net/ipv4/tcp\_mtu\_probing'**

Sets/enables probing on PC

**PC% cat /proc/sys/net/ipv4/tcp\_mtu\_probing'**

Checks the current setting for probing. If "0" or "1" then it is not set for default probing. Has to "=2" to enable probing.

Routing commands on a PC:

**route add –net netaddress netmask mask gw gw\_address**

**route add –net netaddress netmask mask dev iface**

Adds a routing table entry for the network prefix identified by IP address **netaddress** and netmask **mask**. The next-hop is identified by IP address **gw\_address** or by interface **iface**.

**route add –host hostaddress gw gw\_address**

**route add –host hostaddress dev iface**

Adds a host route entry for IP address **hostaddress** with the next-hop identified by IP address **gw\_address** or by interface **iface**

**route add default gw gw\_address**

Sets the default route to gateway with IP address **gw\_address**

**route del –net netaddress netmask mask gw gw\_address**

**route del –host hostaddress gw gw\_address**

**route del default gw gw\_address**

Deletes an existing route from the routing table with specific arguments.

**route -e**

Displays the current routing table with extended fields. The command is identical to the netstat –r command.

**ip route flush table main**

Deletes all entries in the routing table on a PC. Please note that the local interface route(s) need to be added before adding any other static route entries after a flush table command. To add interface route(s) use the ifconfig **interface** down and ifconfig **interface** up (where e.g. interface = eth0)

**ip route flush cache**

deletes all entries in the routing cache

**ip route get IPAddress**

**ip route flush cache IPAddress**

Displays the cached route for **IPAddress** and flushes the cached route entry for **IPAddress**

Commonly used Linux commands:

**ping** **IPAddress**

Pings host with IP address **IPAddress**

**ping** **IPAddress –cnum**

Where **num** is the number of pings you want issued to destination **IPAddress**

**ping** **IPAddress –snum**

Where **num** is the number of data bytes in the ICMP request message you want sent to destination **IPAddress**

**traceroute IPAddress –mxxx -qyyy**

Command used to trace the route between an origin and a destination **IPAddress**, where –m**xxx** indicates the **max TTL value** and –q**yyy** indicates the **number of queries**. E.g. m=2, and q=1.