CCNA / CCNP Routing The Total Guide For all IOS Commands

Break Sequence Key

Enter break sequence command while router isbooting.

**Note: On some ISR G2 you will need to wait untilimage has been decompressed as commands entered in ROMMON do not stick.*

Software	OS	Break Sequence
Hyperterminal	Windows XP, 2000, 98	Ctrl-Break
Kermit	Unix	Ctrl-\l
		Ctrl-\b
MicroPhone Pro	Windows	Ctrl-Break
Minicom	Linux	Ctrl-a f
Putty		Ctrl+Break
ProComm Plus	DOS or Windows	Alt-b
SecureCRT	Windows	Ctrl-Break
Telix	DOS	Ctrl-End
Telnet	N/A	Ctrl-] *then type* **send brk**
Telnet to Cisco	N/A	Ctrl-]
Teraterm	Windows	Alt-b
Terminal	Windows	Break
		Ctrl-Break
Tip	Unix	Ctrl-], then Break or Ctrl-c
		~#
VT 100 Emulation	n/a	F16
Windows NT	Windows	Break-F5
		Shift-F5
		Shift-6 Shift-4 Shift-b (^$B)
Z-TERMINAL	Mac	Command-b
n/a	Break-Out Box	Connect pin 2 (X-mit) to +V for half a second
	Cisco to aux port	Control-Shft-6, then b
	IBM Compatible	Ctrl-Break

Break key sequence simulation is useful if your terminal emulator does not support the break key, or if a bug does not allow your terminal emulator tosend the correct signal.

Note: The hyperterminal under Windows NT hadthis behavior in the past.

Complete these steps to simulate a break keysequence:

1. Connect to the router with these terminalsettings:
   - 1200 baud rate
   - No parity
   - 8 data bits
   - 1 stop bit
   - No flow control

   You no longer see any output on yourscreen, and this is normal.

2. Power cycle (switch off and then on) therouter and press the SPACEBAR for 10-15 seconds in order to generate a signal similar to the break sequence.

3. Disconnect your terminal, and reconnect with a 9600 baud rate. You enter the ROMMonitor mode.

Password Recovery

Ignore NVRAM on boot up:

- rommon 1> **confreg 0x2142**
- rommon 2> **reset**

Tell router to inpsect NVRAM on next boot:

- Router(config)# **config-register 0x2102**

**Note: Older routes require 0x102*

Save Changes:

- Router# **copy run start**

Basic ROMMON Commands

**Variables are case sensitive*

Display ROMMON Varibles:

- rommon 1> **set**

TFTP ROMMON

**Variables are case sensitive*

Set ROMMON TFTP Variables:

- rommon 1> **IP_ADDRESS=***Device_IP*
- rommon 2> **IP_SUBNET_MASK=***Mask*
- rommon 3> **DEFAULT_GATEWAY=***Gateway_IP*
- rommon 4> **TFTP_SERVER=***Server_IP*
- rommon 5> **TFTP_FILE=***File_name*
- rommon 6> **tftpdnl**

Useful Reminders

IOS Shortcut Keys

Navigation

Ctrl-A Moves cursor to begging of line
Ctrl-B Moves 1 character back
Ctrl-C Exit configuration mode or comes out of –more– mode
Ctrl-E Moves cursor to end of line
Ctrl-F Moves 1 character forward **Ctrl-Z** Exits to EXEC Privileged mode
Enter Goes down by single line in –more– mode
Space Goes down by page in –more– mode

Esc-B Moves 1 word back
Esc-F Moves 1 word forward

Display & Altering Commands

UpArrow Displays previous command
DownArrow Displays last command

Ctrl-K Erase all characters from the current position to end of line
Ctrl-N Displays previous command (UpArrow)
Ctrl-P Displays last command (DownArrow)
Ctrl-R Redisplays line
Ctrl-T Swap character with one before it
Ctrl-U Erase current line
Ctrl-W Erase current word
Ctrl-X Erase all characters from the current position to the begging of line

Command Guide

TAB Will fill out the rest of the IOS command or show list of possible commands

Break Sequence Keys

Telnet:

- **Ctrl-]** then type **send brk**

TeraTerm:

- **Alt-B**

Putty:

- **Ctrl-Break**

Abort lookup, ping ect:

- **Ctrl-Shift-6**
- **Ctrl-Shift-6-X**

Router Check List

- Hostname
- Domain name
- Enable Secret Password
- Service Password Encryption
- Line Console Password & logging synchronous
- VTY login Password
- SSH or Telnet Remote access
- AAA Authentication
- User views
- Routing Protocols
- Static Routes
- Default Route
- DHCP
- Relay Agent
- SNMP
- Syslog
- ACL

Piping

Router# *command | pipe_command variable*
append - Append a file
begin - start with the line that match
exclude - don't include lines that match
redirect - sends output to location
tee - sends output to location & displays it
section - filter a section of output

To View files in flash:
Router# **more flash:***file_name*

Ping Results

! - Echo Reply has been received for Echo request
. - Timed out waiting for Echo Reply
U - Destination unreachable
Q - Source quench
M - Could not fragment
? - Unknown packet type
& - TTL expired

To stop Router trying to access TFTP on start-up:

- Router(config)# **no service config**

Type ‘q’ for exit from –more- lines on ASA firewalls

Upgrade AP with tar file (IOS+Web GUI):

- archive download-sw /create-space /overwrite tftp://TFTP_IP/File.tar

EXEC & Privileged EXEC Commands

Basic Commands

To display commands that can be entered:

- **?**

To negate a command:

- **no** *ios_command*

To restore to default settings:

- **default** *ios_command*

EXEC Commands

Elevating to EXEC Privileged Mode

Router> **enable**

Privileged EXEC Commands

De-elevate to EXEC Mode

Router# **disable**

Enter Global Configuration Mode

Router# **configuration terminal**

AAA

Router# **show aaa local user lockout**

- Show locked users

CDP

Router# **show cdp entry** { ***** | *device_name* }

- Device details

Router# **show cdp neighbor**

- Directly connected devices
- Device names
- Local interfaces
- IOS type
- Device type

Router# **show cdp neighbour detail**

- IP address of interface

DHCP

Router# **show ip dhcp binding**

- View DHCP operation

Router# **show ip dhcp server statistics**

- Verify DHCP messages are being sent & received

Router# **show ip dhcp conflict**

- View IP address conflicts

Router# **debug ip dhcp server**

- Verify the router is sending out DHCP requests

Router# **show ip helper-address**

- shows IP helper addresses

Router# **show key chain** {*name_of_chain*}

- View key chains

NAT

Router# **clear ip nat translation ***

- Clear entry before they time out

Router# **show ip nat translations** [**verbose**]

- View NAT Operations

Router# **debug ip nat detailed**

- Debug NAT

Other Commands

Router# **show archive**

- Shows archive

Router# **show arp**

- ARP table

Router# **show ip interface brief**

- Displays interface status

Router# **show clock**

- View current clock settings

Router# **show controllers**

- DTE/DCE info

Router# **show crypto key mypubkey rsa**

- View existing RSA keys

Router# **show frame-relay lmi**

- View LMI statistics

Router# **show frame-relay pvc** [**interface** *interface*] [**dlci**]

- View PVC and traffic info

Router# **show frame-realy map**

- View InARP

Router# **show interfaces**

- Detailed port info

Router# **show ip shh**

- Show SSH settings

Router# **show key chain** [*name_of_chain*]
- Shows key chains, accept & send dates

Router# **show logging**
- View type & level of logging messages sent to Syslog server

Router# **show login** [**failures**]
- Show login information

Router# **show ntp associations**
- NTP information

Router# **show parser view all**
- Summary of all views, *need to be in root view*

Router# **show privilege**
- Show current privilege level

Router# **show running-config**
- Shows running configuration

Router# **show sessions**
- View telnet & SSH sessions

Router# **show startup-config**
- Shows start-up configuration

Router# **show version**
- Displays IOS version, model type and config registration number

Router(config)# **radius-server host** *ip_address* **?**
- Show radius port numbers

Router# **clear counters**

Router# **clear frame-realy inarp**
- Clear InARP

Router# **show process cpu**
- show CPU statistics

Router# **show tech-support**
- TAKES AGES TO DO!
- show massive details
- Should Pipe the output

Debug Commands

Debug Commands

To stop all debugging:
- Router# **undebug all**

Router# **debug ip packet detail**
- Verify DHCP messages are being sent & received

Debugging RIP:
- Router# **debug ip rip**

Basic Configuration

Interfaces & Sub-interfaces

Configuring Router Interfaces:

- Router(config)# **interface** *interface-type interface-number*
- Router(config-if)# **ip address** *ip-address netmask*
- Router(config-if)# **description** *enter-description*
- Router(config-if)# **no shutdown**

Interface range:

- Router(config)# **interface range** *interface_type int_num*/*int_num* **-** *int_num*

Loopback interfaces:

- Router(config)# **interface loopback** *interface_number*

Null Interfaces:
**Can be used to create supernet*

- Router(config)# **ip route** *ip-address subnet_mask* **Null0**

Sub-interfaces (aka Router on a Stick):

- Router(config)# **interface** *interface interface-number.number*
- Router(config-subif)# **encapsulation** *encapsulation-type VLAN-ID*
- Router(config-subif)# **ip address** *IP-address subnet-mask*

Setting Clock Rate:
Router(config-if)# **clock rate** *clock-rate*

Altering bandwidth of interfaces:
Router(config-if)# **bandwidth** *bnadwdith-in-kbps*

Changing Carrier Delay:
Router(config-if)# **carrier-delay** {*seconds* | **msec** *milliseconds*}

CDP

Globally running and removing CDP:

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

Running and removing CDP from interfaces:

- Router(config-if)# **no cdp enable**
- Router(config-if)# **cdp enable**

DHCP & Relay Agents

To disable DHCP:
**DHCP is on by default*

- Router(config)# **no service dhcp**

Excluding IP address:

- Router(config)# **ip dhcp excluded-address** *ip-address* [*high-ip-address*]

Creating a DHCP pool:

- Router(config)# **ip dhcp pool** *pool-name*
- Router(dhcp-config)# **network** *ip-address sub-netmask*
- Router(dhcp-config)# **default-router** *ip-address* [*ip-address2...8*]
- Router(dhcp-config)# **domain-name** *domain-name*
- Router(dhcp-config)# **dns-server** *ip-address* [*ip-address2...8*]
- Router(dhcp-config)# **lease** {*days* [*hours*] [*minutes*] | **infinite**}
- Router(dhcp-config)# **netbios-name-server** *ip-address* [*ip-address2...8*]

For a routers interface to obtain an IP address:

- Router(config-if)# **ip address dhcp**

Configuring a Router as a relay agent:

- Router(config-if)# **ip helper-address** *dhcp-ip-address*

To forward specific protocols type the following:

- Router(config)# **ip forward-protocol udp** [*port-number*]

To debug a DHCP messages using an ACL:

- Router(config)# **access-list** [*extended-range-number*] **permit ip host 0.0.0.0 host 255.255.255.255**
- Router# **debug ip packet detail** [*acl-number*]

Other Commands

Returning to original settings:

- Router# **reload**

Banner messages:

- Router(config)# **banner motd #** *message* **#**

Enabling & Disabling DNS:

- Router(config)# **no ip domain-lookup**
- Router(config)# **ip name-server** *ip_address*

Enabling HTTP server:

- Router(config)# **ip http server**

Entering EXEC Privilege Commands in Global Configuration Mode:

- Router(config)# **do** *exec_privilege_command*

Setting Local Clock:

- Router# **clock set** *hh:mm:ss Day Month Year*

Maintenance

Basic File Management

Copy Running Config to Start-up cofig:

- Router# **copy running-config startup-config**

Backing up configurations to TFTP:

- Router# **copy** { **running-config** | **startup-config**} **tftp**

Removing Configuration files:

- Router# **erase startup-config**

Create a directory:

- Router# **mkdir flash:***File_name*

Router as a TFTP Server

Copy IOS from Router:

- Router_1(config)# **tftp-server flash:***/IOS_Image*
- Router_2# **copy tftp flash**

File Transferring

Assign a Password and Username to service:

- Router(config)# **ip** {**ftp** | **http**} **username** *Username*
- Router(config)# **ip** {**ftp** | **http**} **password** *Username*

Copy to FTP & HTTP with predefined user & p/w:

- Router# **copy** {**startup-config | ...** } {**ftp** | **http**}**:**//*IP_Addess*/*File_name*

Copy to FTP & HTTP with NO predefined user & p/w:

- Router# **copy** {**startup-config | ...** } {**ftp** | **http**}**:**//*Username***:***Password***@***IP_Addess* /*File_name*

Archiving

Archiving:

- Router(config)# **archive**
- Router(config-archive)# **path** {**flash:** | ... }/*directory*/**$h**-config
- Router(config-archive)# **write-memory**
- Router(config-archive)# **time-period** *mins*

**$h means hostname, $t means time stamp*

Logging:

- Router(config-archive)# **log config**
- Router(config-archive-log-cfg)# **notify syslog**
- Router(config-archive-log-cfg)# **hidekeys**
- Router(config-archive-log-cfg)# **logging enable**
- Router(config-archive-log-cfg)# **logging size** *Max_Commands_to_Log*

Syslog Server

Enable time stamps:

- Router(config)# **service timestamp log** [**datetime** [**msec** | **localtime**] | **uptime**]

Send logging to syslog server:

- Router(config)# **logging** *syslog_server_IP*

Configuring log messages to send:

- Router(config)# **logging trap** [**0 - 7**]

Log User information:

- Router(config)# **logging userinfo**

Rollback

Rollback:

- Router# **configure replace** {**flash:** | ... }*directory*/*config_name* [**list**]

Logging

Logging Buffering:

- Router(config)# **logging buffered** *Level*

SNMP

SNMP:

- Router(config)# **snmp-server community** *community_string* **ro**
- Router(config)# **snmp-server community** *community_string* **rw**
- Router(config)# **snmp-server location** *text*
- Router(config)# **snmp-server contact** *text*
- Router(config)# **snmp-server enable traps** *traps*
- Router(config)# **snmp-server ifindex persist**

Event Manager

EEM:

- Router (config)# **event manager applet** *text*
- Router (config-applet)# **event cli pattern** *pattern* **sync no skip no occurs** *number*

- Router (config-applet)# **action** *label* **syslog priority** *level* **msg** *message*

Verifying

Router# **show archive log config** {**all | statistics | user**}

- See what archive configs have been configured

Router# **show archive config** {**flash: | system: |** ...**}** {**flash: | system: |** ...**}**

- View the differences between two configurations

Device Access

Enable Password

Password Privileged EXEC mode & setting privilege level:

- Router(config)# **enable** {**secret** | **password**} [**5 | 7** *encrypted_password*] *password*

Encrypting password display:

- Router(config)# **service password-encryption**

Setting minimum password length:

- Router(config)# **security passwords min-length** *length*

Console & VTY Password

Configuring Console:

- Router(config)# **line console 0**

Configuring VTY:

- Router(config)# **line vty** *num1 numb2*

Setting password:

- Router(config-line)# **password** [**5 | 7** *encrypted_password*] *password*

Setting local authentication:

- Router(config-line)# **login local**

Synchronize unsolicited messages:

- Router(config-line)# **logging synchronous**

AAA Authentication:

- Router(config-line)# **login authentication** *aaa_auth_name*

Usernames

Create a Username login:

- Router(config)# **username** *name* [**privilege** *privilege-level*] {**password | secret**} *password*

Assigning a command to a different privilege level:

- Router(config)# **privilege** *mode* **level** *level command*

Apply Local Username authentication a interface line:

- *Enter line interface*
- Router(config-line)# **login local**

To stop repeated failed connections:

- Router(config)# **login block-for** *seconds* **attempts** *tries* **within** *seconds*
- Router(config)# **login quite-mode access-class** *{acl-name | acl-number}*
- Router(config)# **login delay** *seconds*
- Router(config)# **login on-failure log** [**every** *login_attempts*]
- Router(config)# **login on-success log** [**every** *login_attempts*]

Time limit for each line interface, before logging out:

- *Enter line interface*
- Router(config-line)# **exec-timeout** *time_in_mins*

Views & AAA

Enable AAA:

- Router(config)# **aaa new-model**

To change to another view or root view:

- Router# **enable** { **view** view-name | **root** }
- *For root view the secret password needs to be enabled*

Create a view:

- Router(config)# **parser view** *view-name*
- Router(config-view)# **secret** *password*
- Router(config-view)# **commands** *parser-mode* [**include** | **include-exclusive** | **exclude**] [**all**] [**interface** *interface-ID* | **command**]

Creating a superview:

- Router(config)# **parser view** *view-name* **superview**
- Router(config-view)# **secret** *password*
- Router(config-view)# **view** *view-name*

To define a named list of authentication methods:

- Router(config)# **aaa authentication login** *{***default** | *name} method*

To assign an AAA list to an interface line:

- *Enter a line, ie VTY line*
- Router(config-line)# **aaa login authentication** *name*

Restricting the amount of failed logins:

- Router(config)# **aaa local authentication attempts max-fail** *attempts*

To unlock a locked user:

- Router# **clear aaa local user lockout** {**username** *name* | **all**}

SSH & Telnet Connections

** Need to change default hostname of device and add domain name*

Configuring domain name:

- Router(config)# **ip domain-name** *domain-name*

Removing existing RSA keys:

- Router(config)# **crypto key zeroize rsa**

Creating RSA keys:

- Router(config)# **crypto key generate rsa general-keys modulus** *modulus_size*

Enable VTY inbound SSH sessions:

- Router(config-line)# **login local**
- Router(config-line)# **transport input** {**ssh | telnet | none | all**}

SSH Version:
Router(config)# **ip ssh version** {**1**| **2**}

Time interval for router to wait for SSH / Telnet client to respond in the negotiation phase:

- Router(config)# **ip** {**ssh** | **telnet**} **time-out** *seconds*

Change default SSH & telnet retires:

- Router(config)# **ip** {**ssh** | **telnet**} **authentication-retires** *tries*

Radius

Enable AAA, & assign list to a line interface.

Configure list to use radius server:

- Router(config)# **aaa authentication login** {**default** | *list_name*} **group** {**radius** | **tacacs+**} [*method*]

Point to Radius server:

- Router(config)# **radius-server host** *ip_address* **key** *name_1*

To change port numbers:

- Router(config)# **no radius-server host** *ip_address* **auth-port** *port_num* **acct-port** *port_num*
- Router(config)# **radius-server host** *ip_address* **auth-port** *port_num* **acct-port** *port_num* **key** *name_1*

Basic Network Security

NTP & Clock

Configuring Clock:

- Router# **clock set** *hhmmss Month Day Year*

NTP Client:

- Router(config)# **ntp server** *Server_IP*
- Router(config)# **update-calender**

NTP Master:

- Router(config)# **ntp master** *stratum-num*

Securing Config and IOS

Secure IOS image and Configuration File:

- Router(config)# **secure boot-image**
- Router(config)# **secure boot-config**

ACL Basics

ACL order on packets:
On incoming packets = ACL > NATOn outgoing packets = NAT > ACL

Common ACL Ranges:

ACL	Type
1-99	Standard ACL
100-199	Extend ACL
1100-1199	MAC ACL *(Extended Range)*
1300-1999	Standard ACL *(Extended Range)*
2000-2699	Extended ACL *(Extended Range)*
700-799	MAC ACL

Where to place ACLs:

ACL	ACL Location
Standard	Destination Network
Extended	Source Network

Standard ACLs

Configure Standard ACL:

- Router(config)# **access-list** *ACL-Num* {**permit** | **deny**} *source-addr source-wildcard*

Extended ACLs

Configure Extended ACL:

- Router(config)# **access-list** *ACL-Num* {**permit** | **deny**} *protocol source-addrsource-wildcard* [*operator operand*] *destination-addr destination-wildcard* [*operator operand*] [**established**] [**log**]

Named ACLs

Naming an ACL:

- Router(config)# **ip access-list** {**standard** | **extended**} *name-of-ACL*

Named Standard ACL configuration:

- Router(config-std-nacl)# {**deny** | **permit**} {*source-addr source-wildcard* | ***any***}

Named Extended ALC configuration:

- Router(config-ext-nacl)# {**permit** | **deny**}*protocol source-addr source-wildcard* [*operator operand*] *destination-addr destination-wildcard* [*operator operand*] [**established**]

Reflexive ACLs

Internal reflexive ACL, to check for outboundtraffic:

- Router(config)# **ip access-list extended** *internal-ACL-name*

- Router(config-ext-nacl)# **permit** *protocolsource-addr source-mask* [*operator operand*] *destination-addr destination- mask* [*operator operand*] [**established**] [**log**] **reflect** *reflect-ACL-name* [**timeout** *seconds*]

External ACL, to check inbound traffic:

- Router(config)# **ip access-list extended** *external-ACL*
- Router(config-ext-nacl)# **evaluate** reflexive-ACL-name

To create a dynamic ACL entry:

- Router(config)# **access-list** {**100** – **199**}**dynamic** *dynamic-ACL-name* [**timeout** *minutes*] {**permit** | **deny**} *protocol source-addr source-wildcard* [*operatoroperand*] *desintation-addr destination-wildcard* [*operator operand*] [**established**]

To enable lock-&-key authentication on VTY lines:

- Router(config-line)# **autocommand access-enable host** [**timeout** *minutes*]

To create Time based ACLs:

- Router(config)# **time-range** *time-range-name*
- Router(config-time-range)# **absolute** {*start-time* | *start-date*} {*end-time* | *end-date*}
- Router(config-time-range)# **periodic** *day-of-week hh:mm* **to** [*day-of-week*] *hh:mm*

Creating a time-based ACL:

- Router(config)# **access-list** {**100** – **199**} {**permit** | **deny**} *protocol source-addr source-wildcard* [*operator operand*] *desintation-addr destination-wildcard* [*operator operand*] [**established**] **time-range** *time-range-name*

Apply ACLs

To assign an ACL to an interface:

- Router(config-if)# **ip access-group** {*ACL-Num* | *ACL-Name*} {**in** | **out**}

To assign an ACL to a VTY line:

- Router(config-line)# **access-class**{*ACL-Num* | *ACL-Name*} {**in** | **out**}

IPsec

IPsec Basics

IPsec

Enable IKE:

- Router(config)# **crypto isakmp enable**

Create ISAKMP policy:

**Note: ISAKMP policy must match at other end*

- Router(config)# **crypto isakmp policy** *num*
- Router(config-isakmp)# **authentication pre-share**
- Router(config-isakmp)# **encryption** [**3des** | **des** | **aes**] *num*
- Router(config-isakmp)# **hash** [**sha** | **md5**]
- Router(config-isakmp)# **group** [**1**|**3**|**5**]
- Router(config-isakmp)# **lifetime** *seconds*

Pre-share Keys:

- Router(config)# **crypto isakmp key** *string* **address** *VPN_endpoint_IP*

IPsec transform & lifetimes:

- Router(config)# **crypto ipsec transform-set** *tag_1* **esp-aes 256 esp-sha-hmac**

Change IPsec association lifetimes:

- Router(config)# **crypto ipsec security-association lifetime seconds** *seconds*

Interesting traffic:

- Router(config)# **access-list** *acl_1* **permit ip** *source_IP source_wildcard destination_IP destiantion_wildcard*

Create crypto map:

- Router(config)# **crypto map** *name_1 seq_num* **ipsec-isakmp**
- Router(config-crypto-map)# **match address** *acl_1*
- Router(config-crypto-map)# **set peer** *VPN_endpont_IP*
- Router(config-crypto-map)# **set pfs group**
- Router(config-crypto-map)# **set transform-set** *tag_1*
- Router(config-crypto-map)# **set security-association lifetime seconds** *seconds*

Apply Crypto map to interface:

- Router(config-if)# **crypto map** *name_1*

Verifying IPsec

Router# **show crypto map** [**interface** *int_ID*]

- Displays crypto map specifics

Router# **show crypto session** [**detail**]

- Displays active crypto sessions

Router# **show crypto ipsec sa**

- Displays settings used by SAs.

GRE

GRE is protocol 47

Configure Basic GRE

Configure Basic GRE:
**NOTE: Make sure router knows where to send packets! I.e default route!*

- Router(config)# **interface tunnel** *Tunnel_Num*
- Router(config-if)# **ip address** *IP_Address*
- Router(config-if)# **tunnel source** {*IP_Add* | *int_type int_num*}
- Router(config-if)# **tunnel destination** *IP_Address*

Change GRE tunnel type:
**By default it is IPv4*

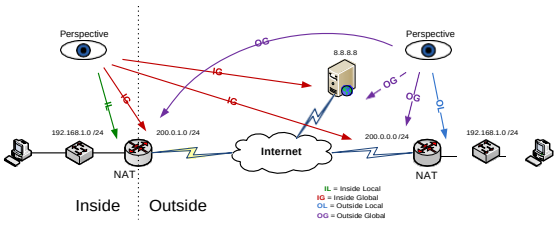
- Router(config-if)# **tunnel mode gre** {**ip** | **ipv6**}

Configure GRE for ACL:

- Router(config-if)# **tunnel mode gre** {**ip** | **ipv6**}

Network Address Translation

NAT order on packets:
On incoming packets = ACL > NAT
On outgoing packets = NAT > ACL



Basic NAT Configuration

There are 4 basic steps for configuring NAT:

1. Configuring inside & outside interfaces
2. Identify IP address to be translated *(using ACLs - NOT used for Static NAT)*
3. Configure Pool *(Dynamic NAT only)*
4. Configure NAT

Configuring inside network:

- Router(config-if)# **ip nat inside**

Configuring outside network:

- Router(config-if)# **ip nat outside**

Static NAT

Static NAT is a one-to-one mapping between devices. This allows a remote device to initiate communication.

Configuring a static NAT map:

- Router(config)# **ip nat inside source static** *local-ip global-ip*

Dynamic NAT

Configuring a pool of global addresses:

- Router(config)# **ip nat** *name start-ip end-ip* **{netmask** *netmask* | **prefix-length** *prefix-length***}**

Define a standard access list:

- Router(config)# **access-list** *number* **permit** *source* [*source-wildcard*]

Establish dynamic source translation:

- Router(config)# **ip nat inside source list** *access-list-number* **pool** *name*

NAT Overload (PAT) with Single IP Address

Define Access List:

- Router(config)# **access-list** *access-list-number* **permit** *source-IP* [*wildcard*]

Configure dynamic source translation:

- Router(config)# **ip nat inside source list** *access-list-number* **interface** *interface* **overload**

To negate dynamic source translation:

- Router(config)# **no ip nat inside source list** *access-list-number* **interface** *interface* **overload**

NAT Overload (PAT) with Multiple IP Address

Configuring global IP address pool:

- Router(config)# **ip nat pool** *name start-ip end-ip* **{netmask** *netmask* | **prefix-length** *prefix-length***}**

Define Access List:

- Router(config)# **access-list** *access-list-number* **permit** *source-IP* [*wildcard*]

Configure overload translation:

- Router(config)# **ip nat inside source list** *access-list* **pool** *name* **overload**

To change time out entries:

- Router(config)# **ip nat translation timeout** *timeout_seconds*

Verify NAT

View NAT Translations:

- Router# **show ip nat** {**translation** | **statistics**}

To clear inside or outside or both dynamic translation entries:

- Router# **clear ip nat translation inside** *global-ip local-ip* [**outside** *local-ip global-ip*]

To clear an extended dynamic translation entry:

- Router# **clear ip nat translation** *protocol* **inside** *global-ip global-port local-ip local-port* [**outside** *local-ip local-port global-ip global-port*]

HDLC & PPP

HDLC & PPP

Configuring HDCL:

- Router(config-if)# **encapsulation hdlc**

Configuring PPP:

- Router(config-if)# **encapsulation PPP**

Configuring Data Compression & quality control:

- Router(config-if)# **compress [predictor | stac]**
- Router(config-if)# **ppp quality** *percentage*

PPPoA

**Need to configure NAT inside interface, ie fa0/0*

Configuring PPPoA ATM Interface & PVC:

- Router(config)# **interface ATM0/0**
- Router(config-if)# **no ip address**
- Router(config-if)# **dsl operting-mode auto**

- Router(config-if)# **pvc** *VPI_num/VCI_num*
- Router(config-if-atm-vc)# **encapsulation** {**aal5mux** | ...} **ppp dialer**
- Router(config-if-atm-vc)# **dialer pool-member** *Dialer_Num_1*

Configuring PPPoA Dialler:

- Router(config)# **interface dialer0**
- Router(config-if)# **ip address** { *IP_add* | **dhcp | negotiated**}
- Router(config-if)# **encapsulation ppp**
- Router(config-if)# **ip nat outside**
- Router(config-if)# **ppp authentication chap callin**
- Router(config-if)# **ppp chap password** *password*

Configure NAT source translation:

- Router(config)# **ip nat inside source list** *ACL_Num* **interface dialer0 overload**
- Router(config)# **access-list** *ACL_Num* **permit ip** *IP_Address* **any**

Default route pointing towards dialler:

- Router(config)# **ip route 0.0.0.0 0.0.0.0 dialer0**

PAP Authentication

**When PAP has authenticated once, it won't authenticate again.*

Configuring PPP PAP:

- Router_1(config)# **username** *User-name-1* **password** *Password-1*
- Router_1(config-if)# **ppp authentication PAP**
- Router_1(config-if)# **ppp pap sent-username** *User-name-2* **password** *Password-2*

- Router_2(config)# **username** *User-name-2* **password** *Password-2*
- Router_2(config-if)# **ppp authentication PAP**
- Router_2(config-if)# **ppp pap sent-username** *User-name-1* **password** *Password-1*

CHAP Authentication

**CHAP re-authenticates at random intervals*

Configuring PPP CHAP:

- Router_1(config)# **username** *Next-Hop-Routers-Name* **password** *Password*
- Router_1(config-if)# **ppp authentication CHAP**

- Router_2(config)# **username** *Next-Hop-Routers-Name* **password** *Password*
- Router_2(config-if)# **ppp authentication CHAP**

Configuring PPP CHAP without Routers hostname:

- Router_1(config)# **username** *User-name-1* **password** *Password*
- Router_1(config-if)# **ppp authentication CHAP**
- Router_1(config-if)# **ppp chap hostname** *User-name-2*

- Router_2(config)# **username** *User-name2* **password** *Password*
- Router_2(config-if)# **ppp authentication CHAP**
- Router_2(config-if)# **ppp chap hostname** *User-name-1*

Frame Relay

Frame Relay - Physical Interface

Configuring Frame Relay:

- Router(config-if)# **ip address** *ip-address subnet-mask*
- Router(config-if)# **encapsulation frame-relay** [**cisco** | **ietf**]

Bandwidth:

- Router(config-if)# **bandwidth** *kbps*

Removing Inverse ARP:

- Router(config-if)# **no frame-relay inverse-arp**

To configure a static DLCI map:
**Disable InARP first*
**Use 'Broadcast' when using routing protocols*

- Router(config-if)# **frame-relay map** *protocol dest-ip-addr local-dlci* [**broadcast** | **cisco** | **ieft**]

To configure a LMI type:
**Configure Keepalive manually as well*

- Router(config-if)# **frame-relay lmi-type** [**cisco** | **ansi** | **q933a]**

Keepalives:
**Default is 10 seconds*

- Router(config-if)# **keepalive** *seconds*

Disable Split Horizon:

- Router(config-if)# **no ip split-horizon**

Frame Relay - Sub Interface

Configure Physical Interface:
**Physical interface needs configuring before sub-interface*
**If LMI needs configuring it can only be done on physical interface*

- Router(config-if)# **encapsulation frame-relay** [**cisco** | **ietf**]
- Router(config-if)# **no ip address**
- Router(config-if)# **no shutdown**

Sub interface:

- Router(config-if)# **interface serial** *number***.** *sub-int number* {**multipoint | point-to-point**}

Configuring DLCI:

- Router(config-subif)# **frame-relay interface-dlci** *DLCI-num*

Bandwidth:

- Router(config-subif)# **bandwidth** *kbps*

Removing Inverse ARP:

- Router(config-subif)# **no frame-relay inverse-arp**

Disable Split Horizon:

- Router(config-subif)# **no ip split-horizon**

Frame Relay Switch

Frame Relay Switch:

- Router(config)# **frame-relay switching**

Frame Relay Switch Static route:

- Router(config-if)# **frame-relay router** *incoming-dlci* **interface** *out-bound-interface out-bound-dlci*

Frame Relay Switch Interface Type:
** Set Clock Rate*

- Router(config-if)# **frame-relay intf-type dce**

Verify Frame Relay

Router# **show interface serial** *number*

- view Encapsulation Type
- LMI Type
- Frame Relay DCE or DTE
- Interface Status

Router# **show frame-relay lmi**

- LMI Type
- Frame Relay DCE or DTE
- LMI sent & received packets

Router# **show frame-relay map**

- Connection Type
- IP to local DLCI mapping
- Broadcasting enabled
- Link status

Router# **show frame-relay pvc** [**interface** *int*] [**dlci**]

- BECN bits
- FECN bits
- PVC Status
- Local DLCI number
- Interface PVC is configured for

Router# **clear frame-relay inarp**

- clear InARP mappings

Basic Routing

Default/Static Routes & Routing behaviour

Default Route:

- Router(config)# **ip route 0.0.0.0 0.0.0.0** *{exit_interface | next-hop_IP_address }*

Classless or Classful Routing behaviour:

- Router(config)# **no ip classless**
- Router(config)# **ip classless**

Static routes:
** Use Exit Interface for point-to-point links.*
**Use Exit interface and Next Hop IP on Broadcast networks*

- Router(config) # **ip route** *ip_address subnet_mask { exit_interface | next-hop_IP_address }*

Permanent Static routes:
** Route will still show in routing table even if Exit interface of next hop IP goes down*

- Router(config) # **ip route** *ip_address subnet_mask { exit_interface | next-hop_IP_address }* [**permanent**]

Changing AD for static Routes:
** Can be used for a backup route*

- Router(config) # **ip route** *ip_address subnet_mask { exit_interface | next-hop_IP_address } A_D*

Creating a Static Null0 Interface/Supernet:
** Used for RIP Supernet Summarization*

- Router(config) # **ip route** *ip_address subnet_mask* **null0**

ODR Routing

Default Route:

- Router(config)# **ip route 0.0.0.0 0.0.0.0** *{exit_interface | next-hop_IP_address }*

Verifying Routing

Router# **show ip route** *IP_Address Subnet* **longer-prefixes**

- Shows Routes with that IP address with a matching subnet mask or greater.

Router(config)# **router ?**

- View routing protocols

Router# **show ip protocols**

- Routing protocols
- Shows AD
- Passive-interfaces

Router# **show ip protocols summary**

- Routing protocol overview

Router# **show ip route**

- All know networks
- Shows AD & metrics

Router# **show ip route summary**

- Shows size of routing table

Path Control

Offset List for Path Control

Extended offset-lists take precedence over normal Offset-Lists.

ALC number 0 means all ACLs. Any IP Route that matches the ACL will have the Offset number applied to it.

Offset Number is what is added to the metric.

Configuring Offset-List:

- Router(config-router)# **offsest-list** *ACL* {**in | out**} *offset_num*

Configuring Extended Offset-List:

- Router(config-router)# **offsest-list** *ACL* {**in | out**} *offset_num* [*int_type int_num*]

IP SLA for Path Control

Configuring IP SLA:

- Router(config)# **ip sla** *operation_number*

  Or, for some IOS images
- Router(config)# **ip sla monitor** *operation_number*

Define IP SLA Operation:

- Router(config-sla-monitor)# **icmp-echo** *Destination_IP* [**source-ip** *Source_IP* [**source-interface** *int_type int_num*]]

  Or, for some IOS images
- Router(config-sla-monitor)# **type echo protocol** ipIcmpEcho *Destination_IP* [**source-ipaddr** *Source_IP* [**source-interface** *int_type int_num*]]

IP SLA Operation additional Commands:

- Router(config-sla-monitor-echo)# **frequency** *seconds*
- Router(config-sla-monitor-echo)# **timeout** *seconds*

Configure Scheduling of IP SLA:

- Router(config)# **ip sla schedule** *operation_number* [**life** {**forever |** *seconds*}] [**start-time** {*hh*:*mm* [:*ss*] [*month day*] **| pending | now | after** *hh*:*mm*:*ss*}] [**ageout** *seconds*] [**recurring**]

  Or, for some IOS images
- Router(config)# **ip sla monitor schedule** ....

IP SLA Object Tracking:

- Router(config)# **track** *object_number* **ip sla** *operation_number* {**state | reachability**}

  Or, for some IOS images
- Router(config)# **track** *object_number* **rtr** *operation_number* {**state | reachability**}

IP SLA Object Tracking Paramers:

- Router(config-track)# {**default delay |** **delay** {**up** *seconds* **| down** *seconds*}}

Action to undertake for Associated Object:

- Router(config)# **ip route** *IP_Address Mask* { *IP_Next_Hop* **|** *int_type int_num* [*IP_Next_Hop*]} [**dhcp [***metric*]] [*A_D*] [**name** *next_hop_name*] [**permanent | track** *object_number*] [**tag** *tag*]

Verifying IP SLA

Router# **show ip sla configuration** [*operation*]

Or, for some IOS images

Router# **show ip sla monitor configuration** [*operation*]

Router# **show ip sla statistst ics** [*operation_number*] [**details**]

Or, for some IOS images

Router# **show ip sla monitor statistics** [*operation_number*] [**details**]

Router# **show track [***object_number*]

PBR for Path Control

Configuring PBR:

**Create either a named or standard ACL.*

- Router(config)# **route-map** *map_name* {**permit | deny}**
- Router(config-route-map)# **match ip address** {*ACL* | **prefix-list** *prefix_list*}
- Router(config-route-map)# **set ip next-hop** *next_hop_IP* [*...next_hop_IP*]

Apply PBR to interface where packets are received:

- Router(config-if)# **ip policy route-map** *map_name*

Allow local originating packets to be PBR:

- Router(config)# **ip local policy route-map** *map_name*

Specify interface in which packets can e sent down:

- Router(config-route-map)# **set interface** *int_type int_num* [*...int_type int_num*]

Default next-hop address:

- Router(config-route-map)# **set ip default next-hop** *next_hop_IP* [*...next_hop_IP*]

Default interface:

- Router(config-route-map)# **set default interface** *int_type int_num* [*...int_type int_num*]

Verifying PBR

Router# **show ip policy**

Router# **show route-map** [*map_name*]

Router# **debug ip policy**

Route Filtering

Route Maps

Configure Route Map:

- Router(config)# **route-map** *name* [**permit | deny**] [*sequence_numb*]
- Router(config-route-map)# **match** {... | **ip** { **address | next-hop | route-source** } { *ACL* | **prefix-list** *prefix_name*}
- Router(config-route-map)# **set** {... | **metric** *metric* }

Distribution Lists

Distribution List with ACL:

- Router(config-router)# **distribute-list** *ACL* {**in | out**} {*int_type int_num* | *routing_protocol*}

Distribution List with Route-Maps:

- Router(config-router)# **distribute-list route-map** *map_tap* {**in | out**} {*int_type int_num* | *routing_protocol*}

Distribution List with Prefix Lists:

- Router(config-router)# **distribute-list prefix** *prefix_name* {**in | out**} {*int_type int_num* | *routing_protocol*}

Prefix Lists

Configure Prefix List:

- Router(config)# **ip prefix-list** {*name* | *list_num*} [**seq** *seq_num*] {**deny | permit**} *IP_add/Subnet_length* [**le** *length* | **ge** *length*] **description** *text*

Prefix List Description:

- Router(config)# **ip prefix-list** {*name* | *list_num*} **description** *text*

Router# **show ip prefix-list** [**detail | summary**]

- Shows all prefix lists

Router# **show ip prefix-list** [**detail | summary**] *prefix-list-name*

- Shows details about the specified prefix list

Router# **show ip prefix-list** *prefix-list-name network/length*

- Shows entry for specific network

Router# **show ip prefix-list** *prefix-list-name*[**seq** *sequence-number*

- Shows entry for given sequence number

Router# **show ip prefix-list** *prefix-list-name* [*network/length*] **longer**

- Shows entries that are more specific than the network length given

Router# **show ip prefix-list** *prefix-list-name* [*network/length*] **first-match**

- Shows the entry that matches the network length

Router# **clear ip prefix-list** *prefix-list-name* [*network/length*]

- Resets the counter

RIP

RIP Basics

RIP Metric:

- Hop count
- Hop count of 16 means network unreadable

Default Timers:

- Update Timer = 30 seconds
- Invalid Timer = 180 seconds
- Hold-down Timer = 180 seconds
- Flush timer = 240 seconds

Configuring RIP

Configuring RIP:

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

Configuring Network addresses for RIP:

- Router(config-router)# **network** *ip_address*

Configuring Passive interfaces:

- Router(config-router)# **passive-interface** *interface_type interface_number*

Default route:

Add default route as normal

- Router(config-router)# **default-information originate**

Static Routes:

Add static route as normal

- Router(config-router)# **redistribute static**

Change RIP versions:

**Need to explicitly say Version 2*

- Router(config-router)# **version 1**
- Router(config-router)# **version 2**

No auto-summary:

- Router(config-router)# **no auto-summary**

Changing timers:

**Values are in seconds*

- Router(config-router)# **timers basic** *Interval Invalid Hold-down Flush*

RIP over Frame Relay

Configuring Passive interfaces globally:

- Router(config-router)# **passive-interface default**
- Router(config-router)# **no passive-interface** *interface*

MD5 Authentication:

- Router(config)# **key chain** ***key-chain-name***
- Router(config-keychain)# **key** *key-ID*
- Router(config-keychain-key)# **key-string** *string*

- Router(config-if)# **ip rip authentication mode md5**
- Router(config-if)# **ip rip authentication key-chain key-chain-name**

Advanced RIP Configuration

Configuring Administrative Distance:

- Router(config-router)# **distance** *A_D Source_IP Wildmask* [*ACL*]

Securing RIP

Configuring Passive interfaces globally:

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

EIGRP

EIGRP Basics

EIGRP Metrics:
- k1 - Bandwidth *(On by default)*
- k2 - Load
- k3 - Delay *(On by default)*
- k4 - Reliability
- k5 - MTU

EIGRP Metric calculation:
Metric = ([k1 * bandwidth + (k2 * bandwidth) / (256 – load) + k3 * delay] * [k5 / (Reliability + k4)])
**k3 is the sum of all delays*

Hello Interval Timers:
- **5** seconds for LAN networks
- **60** seconds for NMBA networks

Hold-Down Timers:
Should be '*x3***' the size of Hello Interval Timer*
- **15** seconds for LAN networks
- **180** seconds for NMBA networks

Active-Timer: 3 minutes default. Used for SIA-Reply and SIA-Query.

Basic EIGRP Configuration

Remove Auto-summary:
- Router(config-router)# **no auto-summary**

Configure EIGRP:
- Router(config)# **router eigrp** *process_id*

Configuring Network addresses for EGRIP:
- Router(config-router)# **network** *network_address* [*wildcard_mask*]

Changing metric values:
- Router(config-router)# **metric weights 0** *k1 k2 k3 k4 k5*

Changing Bandwidth:
**Do not alter for changing the metric. Use delay instead.*
- Router(config-if)# **bandwidth** *kilobits*

Changing Bandwidth Percentage:
- Router(config-if)# **ip bandwidth-percent eigrp** *Process_ID Percentage*

Changing Delay:
- Router(config-if)# **delay** *tens_of_microseconds*

Summary Address:
- Router(config-if)# **ip summary-address eigrp** *process_ID network_address subnet_mask* [*AD*]

Summary Address using Null0:
**Must redistribute*
- Router(config)# **ip route** *ip-address subnet_mask* **Null0**

Redistribute static routes:
**Add static routes as normal*
- Router(config-router)# **redistribute static**

Changing Router ID:
- Router(config-router)# **eigrp router-id** *ip_address*

Default Network:
*Use if you don't want to redistribute static links
- Router(config)# **ip default-network** *IP_address*

EIGRP Timers

Changing 'Hello' & 'Hold Time' intervals:
- Router(config-if)# **ip hello-interval eigrp** *process_ID seconds*
- Router(config-if)# **ip hold-time eigrp** *process_ID seconds*

Change EIGRP Active-timer:
- Router(config-router)# **timers active-time** { *max_time* | **disalbed**}

Advanced EIGRP Configuration

Change number of equal cost paths to use:
**4 is the default*
- Router(config-router)# **maximum-paths** *1-16*

EIGRP Router ID:
- Router(config-router)# **eigrp router-id** *id*

Change traffic loading:
- Router(config-router)# **traffic-share** {**balanced** | **min across-interfaces**}

Unequal load balancing:
- Router(config-router)# **variance** *1-128*

EIGRP Stub:
**Connected & Summary are default*

- Router(config-router)# **eigrp stub** [**receive-only**] [**connected | static | summary | redistributed**]

Changing EIGRP Administrative Distance:

- Router(config-router)# **distance eigrp** *Internal_AD External_AD*

Changing delay of link down notifications from hardware:

- Router(config-if)# **carrier-delay** {*seconds* | **msec** *milliseconds*}

Log EIGRP neighbour adjacency changes:

- Router(config)# **eigrp log-neighbor-changes**

EIGRP over Frame Relay

Configuring Passive interfaces globally:

Set Unicast Communication:

- Router(config-router)# **neighbor** *Neighbour_IP int_typ int_numb*

Removing Split Horizon:

- Router(config-if)# **no ip split-horizon**

Securing EIGRP

Configuring Passive interfaces globally:

- Router(config-router)# **passive-interface default**
- Router(config-router)# no **passive-interface** *interface*

Pseudo Passive interface:
**EIGRP must be advertising to that network*

- Router(config-router)# **distribute-list** ***Number*** **out** *int_type int_num*
- Router(config)# **access-list** ***Number*** **deny deny**

MD5 Authentication:
**Can't use month number for month*

- Router(config)# **key chain** ***key-chain-name***
- Router(config-keychain)# **key** *key-ID*
- Router(config-keychain-key)# **key-string** *string*

- Router(config-keychain-key)# **accept-lifetime** *hh*:*mm*:*ss day month year* [**infinite** | **duration** *seconds*]
- Router(config-keychain-key)# **send-lifetime** *hh*:*mm*:*ss day month year* [**infinite** | **duration** *seconds*]

Or

- Router(config-keychain-key)# **accept-lifetime** *hh*:*mm*:*ss day month year* {*hh*:*mm*:*ss day month year*}
- Router(config-keychain-key)# **send-lifetime** *hh*:*mm*:*ss day month year* {*hh*:*mm*:*ss day month year*}

- Router(config-if)# **ip authentication mode eigrp** *process-ID* **md5**
- Router(config-if)# **ip authentication key-chain eigrp** *process-ID* ***key-chain-name***

Verifying EIGRP

Router# **show ip protocols**

- Shows *K* values
- EIGRP Process ID
- Route filtering on inbound and outbound updates
- Generating or receiving a default route
- number of load balancing paths

Router# **show ip eigrp interfaces**

- Shows EIGRP interfaces

Router# **show ip eigrp interfaces detail**

- Shows Hello Time Interval
- Shows authentication

Router# **show ip eigrp neighbor**

- Shows adjacencies

Router# **show ip eigrp neighbour** *int_type int_numb*

- Shows adjacencies on that interface

Router# **show ip eigrp traffic**

- Shows number of EIGRP packets sent & received

Router# **show ip eigrp topology**

- Shows successors & FS

Router# **show ip eigrp topology all-links**

- Shows all routes learned through EIGRP

Router# **show ip eigrp topology**

- Shows the successors and FS for Routes

Router# **show ip eigrp topology** *ip_add / netmask*

- EIGRP info on that IP address

Router# **show ip route eigrp**

- Displays EIGRP routable networks

Router# **show key chain** [*name_of_chain*]

- View key chains for EIGRP

Troubleshooting

Router# **debug ip eigrp**

- Show what EIGRP is doing

Router# **debug eigrp packets [***Packet_Type***]** [**detail**]

- Shows EIGRP packets

Router# **debug ip eigrp**

- Shows EIGRP packets sent & received on an interface - create large output

Router# **debug ip eigrp summary**

- IP EIGRP summary route processing

Router# **debug eigrp neighbors**

- Displays neighbours discovered and contents of hello packets

Router# **debug eigrp fsm**

- Shows EIGRP FSM

OSPFv2

OSPF Basics

OSPF Metrics:

- Cost (Reference Bandwidth = 10^8 bps)

OSPF Metric calculation:
Metric = (Reference Bandwidth / Interface Bandwidth)

OSPF Hello Interval Timers:

- 10 seconds on LAN networks
- 30 seconds on NBMA networks

OSPF Dead Interval Timers:
Should be 'x4**' the size of Hello Interval Timer*

- 40 seconds on LAN networks
- 120 seconds on NBMA networks

OSPF LSA Types:

LSAs	Generated by?
Type 1 (Router LSA)	All routers
Type 2 (Network LSA)	Designated Router
Type 3 (Summary LSA)	ABR – Summarize routes outside of own area
Type 4 (Summary LSA)	ABR – Summarizes route to ASBR outside of own area
Type 5 (External LSA)	ASBR (Area0) or ABR NSSA
Type 6 (Multicast LSA)	Not Supported on Cisco Routers
Type 7 (NSSA External LSA)	NSSA ASBR
Type 8	Not used
Type 9 - 11	Opaque LSA

OSPF Area Types:

Area Type	LSAs that propagate inside area	LSA Sent From Area 0	LSAs Sent Out to Area 0
Backbone (Area 0)	1, 2, 3, 4, 5	-	-
Normal Area	1, 2, 3, 5	3, 5	3, 4, 5
Stub Area	1, 2, 3, DR	3, DR	3
Totally-Stubby-Area	1, 2, DR	DR	3
NSSA	1, 2, 3, 7	3	3, 4, 5
NSSA-TSA	1, 2, 7, DR	DR	3, 4, 5

**DR = Default Route - sent from the ABR or ASBR*
**Type 7 LSAs get converted to Type 5 LSAs by the ABR.*

OSPF Router Types:

Router Type	Use
ABR (Area Border Router)	Connects two or more areas
ASBR (Autonomous System Border router)	Connects two different autonomous systems
IR (Internal Router)	Internal Area router
BR (Backbone Router)	Router that connects to Area 0

**Cisco defines ABR as a router that connects another Area to Area 0, though technically this isn't the case all the time*

Configuring OSPF

Configure OSPF:

- Router(config)# **router ospf** *process_id*

Configuring Network addresses for OSPF:

- Router(config-router)# **network** *network-address wildcard-mask* **area** *area_id*

Configuring OSPF on interfaces:
**Can be used instead of 'network' command*

- Router(config-if)# **ip ospf** *Process_ID* **area** *Area_ID*

Setting up Router ID:

- Router(config-router)# **router-id** *ip_address*

Changing Reference Cost Bandwidth:

- Router(config-router)# **auto-cost reference-bandwidth** *rf_BW_Mbps*

Changing Interface Default Bandwidth:

- Router(config-if)# **bandwidth** *bandwidth_value*

or

- Router(config-if)# **ip ospf cost** *bandwidth_value*

Changing Interface Priority:

- Router(config-if)# **ip ospf priority** *{0 – 255}*

Changing 'Hello' & 'Dead' Intervals:

- Router(config-if)# **ip ospf hello-interval** *seconds*
- Router(config-if)# **ip ospf dead-interval** *seconds*

Summary Address using Null0:
**Must redistribute*
** See Summarizing IP Ranges in LSA filtering for alternative command*

- Router(config)# **ip route** *ip-address subnet_mask* **Null0**

Redistributing Static Route:
**Add static route as normal*

- Router(config-router)# **redistribute static**

Default Route:
**Add default route as normal*

- Router(config-router)# **default-information originate**

Area 0 & Normal Areas

Enable a network for Area 0:

- Router(config-router)# **network** *network-address wildcard-mask* **area 0**

Enable a network for a Normal Area:

- Router(config-router)# **network** *network-address wildcard-mask* **area** *Area_ID*

Stub Areas

Stub ABR Router:

- ABR_Router(config-router)# **area** *area_ID* **stub**

Stub Internal Router:

- IR_Router(config-router)# **area** *area_ID* **stub**

Totally Stubby Areas

TSA ABR Router:

- ABR_Router(config-router)# **area** *area_ID* **stub no-summary**

TSA Internal Router:

- IR_Router(config-router)# **area** *area_ID* **stub**

NSSA Areas

NSSA ABR Router:

- ABR_Router(config-router)# **area** *area_ID* **nssa**

NSSA Internal Router:

- IR_Router(config-router **area** *area_ID* **nssa**

NSSA ASBR Router:

- ASBR_Router(config-router)# **area** *area_ID* **nssa**

NSSA Default Route:

- ASBR_Router(config-router)# **area** *area_ID* **nssa default-information-originate**

NSSA TSA Areas

NSSA TSA ABR Router:

- ABR_Router(config-router)# **area** *area_ID* **nssa no-summary**

NSSA TSA Internal Router:

- IR_Router(config-router)# **area** *area_ID* **nssa**

NSSA TSA ASBR Router:

- ASBR_Router(config-router)# **area** *area_ID* **nssa**

LSA Filtering

Filtering Type 7 LSAs:
**Stops Type 7 LSAs being generated in that area.*
**Used when the Router connects to Area0, NSSA area and an external routing source*

- ASBR_Router(config-router)# **area** *area_ID* **nssa no-redistribution**

Stop the NSSA ABR converting Type 7 LSA into Type 5 LSAs:
**Used when other areas don't need to know of the external routing source attached to the NSSA*

- ASBR_Router(config-router)# **summary-address** *ip_addresss subnetmask* **not-advertise**

Summarizing IP ranges (LSA Type 3):
**Use only for OSPF learnt Routes*

- Router(config-router)# **area** *area_ID* **range** *IP_Address Subnet_Mask*

Summarizing IP ranges (LSA Type 3):
**Use for External OSPF learnt Routes*

- Router(config-router)# **summary-address** *IP_address Subnet_Mask*

Virtual Link

Configuration of Virtual Link:

- ABR_1(config-router)# **area** *area_ID* **virtual-link** *ABR_2_ROUTER_ID*

- ABR_2(config-router)# **area** *area_ID* **virtual-link** *ABR_1_ROUTER_ID*

OSPF over Frame Relay

Configuring Passive interfaces globally:

Securing OSPF

Configuring Passive interfaces globally:

- Router(config-router)# **passive-interface default**
- Router(config-router)# **no passive-interface** *interface*

Authentication

Plain Text Authentication:
**Password must match on both interfaces*

- Router(config-if)# **ip ospf authentication-key** *password*

Send Plain Text Authentication:

- Router(config-if)# **ip ospf authentication**

Or

- Router(config-router)# **area** *Area_ID* **authentication**

MD5 Authentication:
**Key and Key-String must match*

- Router(config-if)# **ip ospf message-digest-key** *key-ID* **md5** *string*

Send MD5 Authentication:

- Router(config-if)# **ip ospf authentication message-digest**

Or

- Router(config-router)# **area** *area-ID* **authentication message-digest**

Advanced Configuration

Administrative Distance:
- Router(config-router)# **distance** *A_D*

AD for inter, intra & External Routes:
- Router(config-router)# **distance ospf** {**external** | **inter-area** | **intra-area**} *A_D*

Changing the SPF Throttle Timer:
- Router(config-router)# **timers throttle spf** *ms_delay_between_calculations*

Changing the DBD Retransmit Interval:
- Router(config-if)# **ip ospf retransmit-interval** *seconds*

OSPF Transmit Delay:
- Router(config-if)# **ip ospf transmit-delay** *seconds*

Log OSPF Adjacency:
- Router(config-router)# **log-adjacency-changes** [**detail**]

Send Unicast Hello Packets:
**Default for serial links. No DR or BDR is set. Also used to send subnet mask of loopback interface*
- Router(config-if)# **ip ospf network point-to-point**

Send Loopback Subnet instead of /32:
**Also used to send unicast hello packets on NBMA networks*
- Router(config-if)# **ip ospf network point-to-point**

Verifying OSPF

Router# **show ip ospf** [*process_id*]
- Router ID + other info

Router# **show ip ospf database**
- Shows LSAs
- LSA Age, Sequence Number

Router# **show ip ospf interface** [*int_type int_num*]
- Shows Area Interface belong too
- Timer intervals
- Link Cost
- If authentication is being used

Router# **show ip ospf interface brief**
- Brief overview, PID, Areas, Link Cost

Router# **show ip ospf neighbor** [*int_type int_num*] [*neighbor_id*] [**detail**]
- Shows adjacencies

Router# **show ip ospf virtual-links**
- Shows virtual links

Router# **show ip route ospf**
- Displays OSPF routable networks

Troubleshooting

Router# **debug ip OSPF**
- Basic OSPF debugging

Router# **debug ip ospf events**
- Shows OSPF event

Router# **debug ip ospf packet**
- Shows OSPF packets types sent

Router# **debug ip ospf adj**
- Shows OSPF adjacency events

Reset OSPF Process:
- Router# **clear ip ospf process**

Reset OSPF counters:
- Router# **clear ip ospf counters**

IS-IS

Installing OSPF:

- Router(config)# **router ospf** *process_id*

Installing an Interface:

- Router(config-router)# **network** *network-address wildcard-mask* **area** *area_id*

Setting up Router ID:

- Router(config-router)# **router-id** *ip_address*

To recalculate Router ID:

- Router# **clear ip ospf process**

Changing Reference Cost Bandwidth:

- Router(config-router)# **auto-cost reference-bandwidth** *reference_bandwidth_in_Mbps*

BGP

BGP Basics

OSPF Metrics:

- Cost (Reference Bandwidth = 10^8 bps)

Basic BGP Configuring

To configure BGP:
** Only 1 instance of BGP can run on a router.*

- Router(config)# **router bgp** *AS_Number*

To configure a neighbour:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* }**remote-as** *AS_Number*

To shutdown a BGP neighbour:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **shutdown**

Using a loopback as source IP address:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **update-source loopback** *int_num*

EBGP Multi-hop:
**Create a Static route to the loopback pointing to the physical interface*

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **ebgp-multihop** *ttl_value*

Change the Next-Hop Attribute to self router:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **next-hop-self**

To turn synchronization off type the following:

- Router(config-router)# **no synchronization**

Change Router ID:

- Router(config-router)# **bgp router-id** *IP_ID*

To advertise a network:

- Router(config-router)# **network** *IP_address* [**mask** *subnet_mask*] [**route-map** *map*]

To summarize routes in a routing table:

- Router(config)# **ip route** *IP_address Subnet_Mask* **null0**

BGP authentication:

- Router(confog-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **password** *string*

Resetting BGP

Hard Reset:

- Router# **clear ip bgp** {***** | *neighbor_IP*}

Soft Reset for Inbound & Outbound:

- Router# **clear ip bgp soft**

Soft Reset for Outbound Policy:

- Router# **clear ip bgp** {***** | *neighbor_IP* } **soft out**

Save neighbour updates for soft reset inbound policy:

- Router(config-router)# **neighbor** {***** | *neighbor_IP* } **soft-reconfiguration**
- Router# **clear ip bgp** {***** | *neighbor_IP* } **soft in**

Ensure route refresh is enabled:

- Router# **show ip bgp neigbors**

**Output: Received route refresh capability from peer.*

Dynamic route refresh:

- Router# **clear ip bgp** {***** | *neighbor_IP* } **in**

BGP Peer Grouping

To create a peer-group:

- Router(config-router)# **neighbor** *peer_group_name* **peer-group**

To assign neighbours to a peer group:

- Router(config-router)# **neighbor** *IP_Addres* **peer-group** *peer_group_name*

To rest connection for peer groups:

- Router# **clear ip bgp peer-group** *peer_group_name*

BGP Route-Maps

See section about route-maps to show how to use the commands.

Match Parameters	Set Parameters
match as-path	set as-path
match community	set clns
match clns	set automatic-tag
match interface	set community
match ip address	set interface
match ip next-hop	set default interface
match ip route-source	set ip default next-hop
match metric	set level
match route-type	set local-preference
match tag	set metric
	set metric-type
	set next-hop
	set origin
	set tag
	set weight

BGP Route Manipulation

Router to compare the MED to a network via different AS:

- Router(config-router)# **bgp always-compare-med**

IEFT Missing MED value:

- Router(config-router)# **bgp bestpath med missing-as-worst**

Stop a BGP router decice route based on AS-Path:

- Router(config-router)# **bgp bestpath as-path ignore**

BGP weight Attribute can be changed using:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **weight** *weight*

To change Local Preference for ALL routes type:

- Router(config-router)# **bgp default local-preference** *Preference*

Extending AS path type (Prepending):

- Router(config)# **route-map** *Route_Map_Name* **permit** *Number*
- Router(config-route-map)# **set as-path** {**tag** | **prepend** *AS_Num_1 AS_Num...*}
- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **route-map** *Route_Map_Name* **out**

To change MED type:

- Router(config-router)# **default-metric Metric**

Changing MED with Route Map:

- Router(config)# **route-map** *Route_Map_Name* **permit** *Number*
- Router(config-route-map)# **set metric** *Metric*
- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **route-map** *Route_Map_Name* **out**

Filter-List type:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **filter-list** *ACL* {**out | in**}

Prefix-List type:

- Router(config-router)# **neighbor** { *IP_Addres* | *peer_group_name* } **prefix-list** *Prefix-List* {**out | in**}

Verifying BGP

Router# **show ip bgp**

- Display BGP Table

Router# **show ip bgp summary**

- Status of all BGP connections

Router# **show ip bgp neighbors**

- TCP & BGP Connection

Router# **show ip bgp rib-failure**

- BGP routes that were not installed into the routing table & why

Router# s**how ip bgp neighbors** *IP_Add* **routes**

- Show Accepted neighbour routes

Router# **shows ip bgp neighbors** *IP_Add* **received-routes**

- Show both Accepted and Rejected neighbour routes

Router# **show ip bgp neighbors** *IP_Add* **advertise-routes**

- Show BGP routes advertised to neighbours

Troubleshooting BGP

To observe the BGP states, type the following:

- Router# **debug ip bgp ipv4 unicast [events**]

Other useful debug options:

- Router# **debug ip bgp dampening**
- Router# **debug ip bgp events**
- Router# **debug ip bgp keepalives**
- Router# **debug ip bgp updates**

IPv6

IPv6 Basics

Address Ranges

Address	Description
::/0	Default Route
::1	Local Loopback
2000::/3	Global Unicast
2001:0::/32	Teredo
2002::/16	6to4
FE80::/10	Link-Local
FF00::/8	Multicast
x:x:x:x:0000:5EFE::/64	ISATAP

Multicast Ranges

Address	Description
FF02::1	All nodes on link
FF02::2	All routers on link
FF02::5	OSPF
FF02::6	OSPF DR
FF02::9	RIP
FF02::a	EIGRP
FF02::1:FFxx:xxxx	Solicited Node Multicast. X's are the far right 24bits of unicast or anycast address
FF05::1:3	All DHCP Servers
FF05::101	All NTP Servers

Basic IPv6 Configuration

IPv6 on an interface:
- Router(config-if)# **ipv6 address** *address/prefix* [**eui-64** | **anycast**]

EUI-64 on an interface type:
- Router(config-if)# **ipv6 address** *x:x:x:x*:**:** / **64 eui-64**

IPv6 Anycast Address:
- Router(config-if)# **ipv6 address** *address/prefix* **anycast**

Static Link-Local Address:
- Router(config-if)# **ipv6 address** *address* **link-local**

Static Global Address:
- Router(config-if)# **ipv6 address** *address/prefix*

Unnumbered IPv6:
- Router(config-if)# **ipv6 unnumbered** *Int_type Int_num*

Stateless IPv6 Configuration:
- Router(config-if)# **ipv6 address autoconfig** [**default**]

To enable CEF for IPv6:
- Router(config)# **ipv6 cef**

Time that IPv6 Node is considered reachable:
- Router(config-if)# **ipv6 nd reachable-time** *milliseconds*

**Default is 0, meaning unspecified time.*

Statically map a neighbours IPv6 to MAC:
- Router(config)# **ipv6 neighbor** *IPv6_Add Int_type Int_num MAC_Add*

Display IPv6 Interfaces:
- Router# **show ipv6 interface** {*Int_type Int_num* [**prefix**] | **brief**}

View IPv6 local reachable routers:
- Router# **show ipv6 routers** [*Int_type Int_num*] [**conflicts**]

View IPv6 Neighbours:
- Router# **show ipv6 neighbors** [Int_type Int_num | IPv6_Add | IPv6_Hostname | **statistics**]

Show the details of an IPv6 interface type:
- Router# **show ipv6 interface** {*Int_type Int_num* | **brief**}

Debug IPv6:
- Router# **debug ipv6 nd**
- Router# **debug ipv6 packet** [**access-list** *ACL* | **detail**]

IPv6 Routing

Allow IPv6 Forwarding:
- Router(config)# **ipv6 unicast-routing**

Static Routing:
- Router(config)# **ipv6 route** *IPv6_address/prefix* { *IPv6_address* | *Int_type Int_Num* [*IPv6_address*]} [*Admin_Distance*] [*Admin_Multicast_Distance* | **unicast** | **multicast**] [*next_hop_IPv6*] [**tag** *tag*]

Default Route:
- Router(config)# **ipv6 route ::/0** { *IPv6_address* | *Int_type Int_Num IPv6_address*}

Display IPv6 Routing Table:
- Router# **show ipv6 route** [*IPv6_address* | *prefix* | *protocol* | *Int_type Int_Num* | **static**]

Display IPv6 Routing Protocols:

- Router# **show ipv6 protocols** [**summary**]

RIPng

Enable RIPng from Global Configuration Mode:

- Router(config)# **ipv6 router rip** *RIP_Name*

Enable RIPng from Interface:

- Router(config-if)# **ipv6 rip** *RIP_Name* **enable**

Disable Split horizon:

- Router(config-rtr)# **no split-horizon**

Change Port and Multicast Address:

- Router(config-rtr)# **port** *Port_Num* **multicast-group** *Multicast_IPv6*

Redistribute between RIP processes:

- Router(config-rtr)# **redistribute rip** *RIP_Name*

Debug RIPng:

- Router# **debug ipv6 rip** [*Int_type Int_Num*]

OSPFv3

Enable OSPFv3 from Global Configuration Mode:

- Router(config)# **ipv6 router ospf** *process*

Enable OSPFv3 from Interface:

- Router(config-if)# **ipv6 ospf** *process* **area** *Area_ID* [**instance** *Instance_Id*]

Define Router ID via OSPF:

- Router(config-rtr)# **router-id** *ID*

OSPF Priority:

- Router(config-if)# **ipv6 ospf priority** *value*

OSPF cost on Interface:

- Router(config-if)# **ipv6 ospf cost** *interface_cost*

Summarize routes:

- Router(config-rtr)# **area** *ID* **range** *IPv6_address/prefix* [**advertise** | **non-advertise**] [**cost** *cost*]

Clear OSPF Process:

- Router# **clear ipv6 ospf** [*process_ID*] {**process | force-spf | redistribution | counters | neighbor** [*neighbor_interface | neighbor_ID*]}

Display General OSPFv3 information:

- Router# **show ipv6 ospf** [*process_ID*] [*Area_ID*]

Display OSPFv3 neighbour information:

- Router# **show ipv6 ospf** [*process_ID*] [*Area_ID*] **neighbor** [*Int_Type Int_Num*] [*neighbor_ID*] [**detail**]

Display OSPFv3 interface information:

- Router# **show ipv6 ospf** [*process_ID*] [*Area_ID*] **interface** [*Int_Type Int_Num*] [**brief**]

Debug OSPFv3:

- Router# **debug ipv6 ospf** {**packet | ...**}

EIGRP

Enable EIGRP from Global Configuration Mode:

- Router(config)# **ipv6 router eigrp** *process_ID*

Bring EIGRP process up:

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

Enable EIGRP from Interface:

- Router(config-if)# **ipv6 eigrp** *process_ID*

Define Router ID via EIGRP:

- Router(config-rtr)# **router-id** *ID*

Summarize routes:

- Router(config-if)# **ipv6 summary-address eigrp** *process_ID IPv6_Add* [*Admin_Distance*]

Display EIGRP neighbour information:

- Router# **show ipv6 eigrp neighbor** [*process_ID*] [*Int_Type Int_Num*] [*neighbor_ID*]

Debug EIGRP:

- Router# **debug ipv6 eigrp**

MBGP

Configure MBGP:

- Router(config)# **router bgp** *AS_Num*

MBGP Router ID:

- Router(config-router)# **bgp router-id** *ID*

MBGP Neighbour:

- Router(config-router)# **neighbor** {*IPv6_Add* | *peer_name*} **remote-as** *AS_Num*

IPv6 Routable Addresses:

- Router(config-router)# **address-family ipv6** [**unicast | multicast | vpnv6**]
- Router(config-router-af)# **neigbor** *IPv6_Add* **active**

Advertise IPv6 networks:

- Router(config-router-af)# **network** *IPv6_Network*

IPv6 Route Map:

- Router(config-router-af)# **neighbor** *IPv6_Add* **route-map** *name* {**in | out**}

Tunnelling

Manual Tunnel:

- Router(config)# **interface tunnel** *num*
- Router(config-if)# **ipv6 address** *IPv6_Add*
- Router(config-if)# **tunnel source** *Int_type Int_Num*
- Router(config-if)# **tunnel destination** *IP_add*
- Router(config-if)# **tunnel mode ipv6ip**

GRE Tunnel Mode:

- Router(config-if)# **tunnel mode gre** {**ip | ipv6**}

6to4 Tunnel Mode:

- Router(config-if)# **tunnel mode ipv6ip 6to4**

ISATAP Tunnel Mode:

- Router(config-if)# **tunnel mode ipv6ip isatap**

ISATAP EUI-64 Format Tunnel Address:

- Router(config-if)# **ipv6 address** *IPv6_Add*/**64 eui-64**

Display Tunnel States:

- Router# **show interface tunnel** *number*

IPv6 Translation

Static NAT:

Configure Prefix:

- Router(config)# **ipv6 nat prefix** *IPv6_Add*/**96**

IPv4 to IPv6 Static Translation:

- Router(config)# **ipv6 nat v4v6 source** *IPv4_Add IPv6_Add*

IPv6 to IPv4 Static Translation:

- Router(config)# **ipv6 nat v6v4 source** *IPv6_Add IPv4_Add*

Display NAT-PT Translations:

- Router# **show ipv6 nat translation**

Dynamic NAT:

IPv4 to IPv6 Dynamic Translation:

- Router(config)# **ipv6 nat v4v6 source** {**list** {*ACL* | *name*} **pool** *name*}

IPv6 to IPv4 Dynamic Translation:

- Router(config)# **ipv6 nat v6v4 source** {**list** {*ACL*} **pool** *name*}

IPv6 Pool:

- Router(config)# **ipv6 nat v4v6 pool** *name Start_IPv6 End_IPv6* **prefix-length** *prefix*

IPv4 Pool:

- Router(config)# **ipv6 nat v6v4 pool** *name Start_IPv4 End_IPv4* **prefix-length** *prefix*