Access Control Lists for Securing Traffic

Set ACL in global config, then apply to interfaces.
Put the right ACL on the right port! Outbound or inbound?
Put the ACL on the right device!
- standard ACLs should be close to destination, and extended ACLs close to source
- Prevent unecessary network traffic that will just be denied by the subnet on the other side.
- Use as-needed granularity on the client end prevent a mass denial of traffic.

Corp(config)#access-list {1-99 | 1300-1999} - for standard
Corp(config)#access-list {100-199 | 2000-2699} - for IP extended access list

None of these source addresses should be ever be allowed to enter a production network:
Deny any source addresses from your internal networks.
Deny any local host (127.0.0.0/8) and reserved private addresses (RFC 1918).
Deny any addresses in the IP multicast address range (224.0.0.0/4).

Put most specific (single IPs) first. Subnet or range of subnets lower on list
Put permits first then place deny entries lower on the lists to filter properly
e.g.: Place 10.1.1.1 allow first, and have a later entry for deny 10.1.1.x for the rest of that subnet.

After adding restrictions (deny statements) append an explicit "permit any" at end of rules to counteract implicit deny. See the set ACLs listed in the running-config
You need to disable an ACL before making changes: no ip access-list 3

Corp(config)#access-list 10 [deny | permit | remark] [any | subnet | host] [wildcard (for subnets)]
Wildcard masks: for a block of addresses. Each block size must start at 0 or a multiple of the block size.

Corp(config)#access-list 10 deny 172.16.0.0 0.0.255.255
Match the first two octets and that the last two octets can be any value
Corp(config)#access-list 10 deny 172.16.16.0 0.0.3.255
This configuration tells the router to start at network 172.16.16.0 and use a block size of 4
The range would then be 172.16.16.0 through 172.16.19.255
Corp(config)#access-list 10 deny 172.16.16.0 0.0.7.255
172.16.16.0 going up a block size of 8 to 172.16.23.255
Corp(config)#access-list 10 deny 172.16.32.0 0.0.15.255
172.16.32.0 and goes up a block size of 16 to 172.16.47.255
Corp(config)#access-list 10 deny 172.16.64.0 0.0.63.255
172.16.64.0 and goes up a block size of 64 to 172.16.127.255
Corp(config)#access-list 10 deny 192.168.160.0 0.0.31.255
192.168.160.0 and goes up a block size of 32 to 192.168.191.255

Router(config)#access-list 1 deny 172.16.128.0 0.0.31.255 --- /19 is a 32 block means 144 is in 128-160.0
Router(config)#access-list 1 deny 172.16.48.0 0.0.15.255 --- /20 is a 16 block means 50 is in 48-60.0
Router(config)#access-list 1 deny 172.16.192.0 0.0.63.255 ---/18 is a 64 block means 198 is in 192-254.0
Router(config)#access-list 1 deny 172.16.88.0 0.0.7.255 ---/21 is an 8 block means 92 is in 88-106.0
Router(config)#access-list 1 permit any
Router(config)#interface serial 0
Router(config-if)#ip access-group 1 out
It turns out the rules could have been done this with one line:
Router(config)#access-list 1 deny 172.16.0.0 0.0.255.255
The value of 0.0.0.0/255.255.255.255 can be specified as any.

Need: a standard access list that permits only the host or hosts you want to be able to telnet into the routers, then apply the access list to the VTY line with the access-class in command.
Lab_A(config)#access-list 50 permit host 172.16.10.3
Lab_A(config)#line vty 0 4
Lab_A(config-line)#access-class 50 in <--- use "-class" when configuring lines instead of "-group"

Users on the Sales LAN (172.16.40.0/24 - Fa0/0) should not have access to the Finance LAN (172.16.50.0/24 Fa0/1), but they should be able to access the Internet and the marketing department files. The Marketing LAN needs to access the Finance LAN for application services.
Lab_A(config)#access-list 10 deny 172.16.40.0 0.0.0.255
Lab_A(config)#access-list 10 permit any
Place it close to the source
If you place it as an incoming access list on Fa0/0, you might as well shut down the interface because all of the Sales LAN devices will be denied access to all networks attached to the router. The best place to apply this access list is on the Fa0/1 interface as an outbound list:
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group 10 out
---
access-list 165 [deny | permit | remark] [protocol] {sourceIP maskWC} {destIP maskWC} [logical op] [port | tos]
For a single host, use "host" before the IP address and no wildcard mask is required

Extended ACLs
interface Serial 0/0
Access-group 100 in
access-list 100 remark Begin -- Allow BGP IN and OUT
access-list 100 permit tcp host 1.1.1.1 host 2.2.2.2 eq bgp
access-list 100 permit udp any host 2.2.2.2 gt 33000
access-list 100 remark End
access-list 100 deny ip any any log

interface BRI 0/0
Access-group 100 in
Access-group 100 out

access-list 100 permit ip any any log
debug IP packet detail XXX (access list number)

access-list 101 permit tcp any any eq telnet
debug ip packet detail 101
[IP packet debugging is on (detailed) for access list 101]

Corp(config)#access-list 110 deny tcp any host 172.16.30.2 eq 23 log

This line says to deny any source host trying to telnet to destination host 172.16.30.2. Keep in mind that the next line is an implicit deny by default. If you apply this access list to an interface, you might as well just shut the interface down because by default, there's an implicit deny all at the end of every access list. *So we've got to follow up the access list with the following command:*
Corp(config)#access-list 110 permit ip any any

As always, once our access list is created, we must apply it to an interface with the same command used for the IP standard list:
Corp(config-if)#ip access-group 110 in
Or this:
Corp(config-if)#ip access-group 110 out

Extended ACLs #2
Lab_A#config t
Lab_A(config)#access-list 110 deny tcp any host 172.16.50.5 eq 21
Lab_A(config)#access-list 110 deny tcp any host 172.16.50.5 eq 23
Lab_A(config)#access-list 110 permit ip any any
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group 110 out

Router(config)#access-list 110 deny tcp any 172.16.48.0 0.0.15.255 eq 23
Router(config)#access-list 110 deny tcp any 172.16.192.0 0.0.63.255 eq 23
Router(config)#access-list 110 permit ip any any
Router(config)#interface Ethernet 1
Router(config-if)#ip access-group 110 out
Router(config-if)#interface Ethernet 2
Router(config-if)#ip access-group 110 out

Allow HTTP access to the Finance server from source Host B only. All other traffic will be permitted. We need to be able to configure this in only three test statements:
Lab_A(config)#access-list 110 permit tcp host 192.168.177.2 host 172.22.89.26 eq 80
Lab_A(config)#access-list 110 deny tcp any host 172.22.89.26 eq 80
Lab_A(config)#access-list 110 permit ip any any
Lab_A(config)#interface fastethernet 0/1
Lab_A(config-if)#ip access-group 110 out
webserver:
access-list 253 permit tcp 127.16.5.0 0.0.0.255 eq www any <--permit from 172.16.5.* of www to anyone

Named Lists - these start with "ip access-"
Lab_A(config)#ip access-list standard BlockSales
Lab_A(config-std-nacl)#deny 172.16.40.0 0.0.0.255
Lab_A(config-std-nacl)#permit any
Lab_A(config-std-nacl)#exit
Lab_A(config)#int fa0/1
Lab_A(config-if)#ip access-group BlockSales out

Adding #comments#
R2(config)#access-list 110 remark Permit Bob from Sales Only To Finance
R2(config)#access-list 110 permit ip host 172.16.40.1 172.16.50.0 0.0.0.255
R2(config)#access-list 110 deny ip 172.16.40.0 0.0.0.255 172.16.50.0 0.0.0.255
R2(config)#ip access-list extended No_Telnet
R2(config-ext-nacl)#remark Deny all of Sales from Telnetting to Marketing
R2(config-ext-nacl)#deny tcp 172.16.40.0 0.0.0.255 172.16.60.0 0.0.0.255 eq 23
R2(config-ext-nacl)#permit ip any any

Adding a line to the sequence list:
Lab_A(config)#do show access-list
Extended IP access list 110
    10 deny tcp any host 172.16.30.5 eq ftp
    20 deny tcp any host 172.16.30.5 eq telnet
    30 permit ip any any
    40 permit tcp host 192.168.177.2 host 172.22.89.26 eq www
    50 deny tcp any host 172.22.89.26 eq www
Lab_A (config)#ip access-list extended 110
Lab_A (config-ext-nacl)#21 deny udp any host 172.16.30.5 eq 69

show access-list - displays all access lists and parameters. Shows stats on usage. Doesn't show interfaces.
show access-list 110 - Reveals parameters for 110. Doesn't show interfaces.
show ip access-list - Shows only the IP access lists configured on the router.
show ip interface - Displays which interfaces have access lists set on them.
show running-config - Shows the access lists and the specific interfaces that have ACLs applied on them

Remember: You can’t have two lists on the same interface, in the same direction- one will overwrite the other. configuration.

This extended ACL is used to permit traffic on the 10.1.1.x network (inside) and to receive ping responses from the outside while it prevents unsolicited pings from people outside, permitting all other traffic.

```
interface Ethernet0/1
ip address 172.16.1.2 255.255.255.0
ip access-group 101 in
access-list 101 deny icmp any 10.1.1.0 0.0.0.255 echo
access-list 101 permit ip any 10.1.1.0 0.0.0.255
```

Note: Some applications such as network management require pings for a keepalive function. If this is the case, you might wish to limit blocking inbound pings or be more granular in permitted/denied IPs.

DHCP Setup on Cisco Devices

Switch(config)#service dhcp -- makes sure it's enabled - "no service dhcp" turns it off
Switch(config)#ip dhcp pool Sales_Wireless --create the pool
Switch(dhcp-config)#network 192.168.10.0 255.255.255.0 -- netID and mask for pool
Switch(dhcp-config)#default-router 192.168.10.1 --default gateway
Switch(dhcp-config)#dns-server 4.4.4.4
Switch(dhcp-config)#lease 3 12 15 --days, hours, minutes - default 24hours
Switch(dhcp-config)#exit
Switch(config)#ip dhcp excluded-address 192.168.10.1 192.168.10.10

DHCP Relay
If you need to provide addresses from a DHCP server to hosts that aren't on the same LAN as the DHCP server, you can configure your router interface to relay DHCP client requests.
Router(config)#interface fa0/0
Router(config-if)#ip helper-address 10.10.10.254

The router and fa0/0 are on 192.168.10.1. 10.10.10.254 is obviously on a different subnet, but is the DHCP server. ip helper-address forwards more than just DHCP client requests.

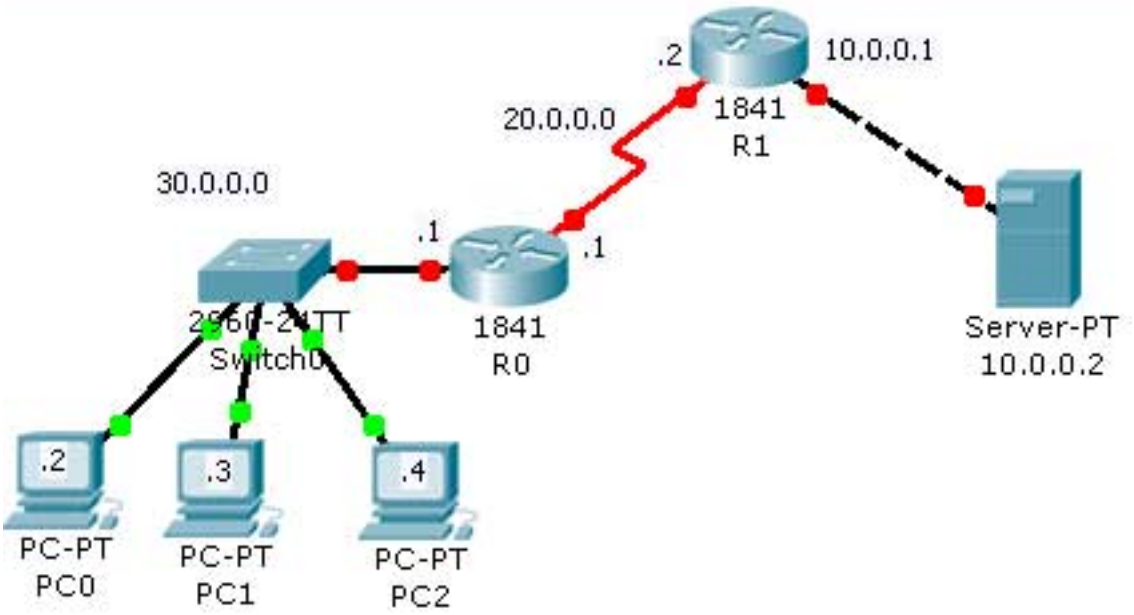
Verifying DHCP
show ip dhcp binding - Lists state information about each IP address currently leased
show ip dhcp pool [poolname] - Lists scope of addresses, plus stats for leased addresses
show ip dhcp server statistics - Lists DHCP server statistics
show ip dhcp conflict - show duplicate addresses

More Pools Example
Corp#config t
Corp(config)#ip dhcp excluded-address 192.168.10.1
Corp(config)#ip dhcp excluded-address 192.168.20.1
Corp(config)#ip dhcp pool SF_LAN
Corp(dhcp-config)#network 192.168.10.0 255.255.255.0
Corp(dhcp-config)#default-router 192.168.10.1
Corp(dhcp-config)#dns-server 4.4.4.4
Corp(dhcp-config)#exit
Corp(config)#ip dhcp pool LA_LAN
Corp(dhcp-config)#network 192.168.20.0 255.255.255.0
Corp(dhcp-config)#default-router 192.168.20.1
Corp(dhcp-config)#dns-server 4.4.4.4

Add these to the two routers so they will forward DHCP:
LA(config)#int f0/0
LA(config-if)#ip helper-address 172.16.10.5

SF(config)#int f0/0
SF(config-if)#ip helper-address 172.16.10.1

Network Address Translation : Static and Dynamic NAT and PAT



IPv4 Static NAT

R1(config-if)#ip address 10.0.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip address 20.0.0.2 255.0.0.0
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#ip route 30.0.0.0 255.0.0.0 20.0.0.1
R1(config)#ip nat inside source static 10.0.0.2 50.0.0.1
R1(config)#interface fastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit

R0(config)#interface fastethernet 0/0
R0(config-if)#ip address 30.0.0.1 255.0.0.0
R0(config-if)#no shutdown
R0(config-if)#exit
R0(config)#interface serial 0/0/0
R0(config-if)#ip address 20.0.0.1 255.0.0.0
R0(config-if)#clock rate 64000
R0(config-if)#bandwidth 64
R0(config-if)#no shutdown
R0(config-if)#exit
R0(config)#ip route 50.0.0.0 255.0.0.0 20.0.0.2

There is not direct route for 10.0.0.2. So PC from network of 30.0.0.0 will never know about it. They will access 50.0.0.1 as the web server IP. Ping 50.0.0.1 and it works. Ping 10.0.0.2 and you will get destination host unreachable error.

IP4 Dynamic NAT
This is using 192.168.0.0 network as internal. We have five public ip address 50.0.0.1 to 50.0.0.5 to use. Router1(1841 Router0) is going to be NAT device. Starting off by configuring Router1(1841 Router0):

R1(config)#interface fastethernet 0/0
R1(config-if)#ip address 192.168.0.1 255.0.0.0
R1(config-if)#no shutdown
R1(config)#interface serial 0/0/0
R1(config-if)#ip address 30.0.0.1 255.0.0.0
R1(config-if)#clock rate 64000
R1(config-if)#bandwidth 64
R1(config-if)#no shutdown
R1(config-if)#exit
R1(config)#ip route 0.0.0.0 0.0.0.0 serial 0/0/0
R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
R1(config)#ip nat pool test 50.0.0.1 50.0.0.5 netmask 255.0.0.0
R1(config)#ip nat inside source list 1 pool test
R1(config)#interface fastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit

R1#debug ip nat
IP NAT debugging is on
NAT: s=192.168.0.7->50.0.0.1, d=20.0.0.2[1]
NAT*: s=20.0.0.2, d=50.0.0.1->192.168.0.7[1]
R1#no debug ip nat
IP NAT debugging is off

As you can see in output 192.168.0.5 is translated with 50.0.0.1 before leaving the router
Webpage loads as well.

IP4 PAT
If you have few global IP address and hundred of inside local address to translate. In such a situation you need to use PAT. This time we are using only one global IP address 50.0.0.1

R1(config)#access-list 1 permit 192.168.0.0 0.0.0.255
R1(config)#ip nat pool test 50.0.0.1 50.0.0.1 netmask 255.0.0.0
R1(config)#ip nat inside source list 1 pool test overload
R1(config)#interface fastEthernet 0/0
R1(config-if)#ip nat inside
R1(config-if)#exit
R1(config)#interface serial 0/0/0
R1(config-if)#ip nat outside
R1(config-if)#exit

To verify PAT go on R1 and run show ip nat translations
R1#show ip nat translations

Pro	Inside global	Inside local	Outside local	Outside global
icmp	50.0.0.1:1	192.168.0.7:1	20.0.0.2:1	20.0.0.2:1
icmp	50.0.0.1:2	192.168.0.7:2	20.0.0.2:2	20.0.0.2:2

IPv6 Introduction

IPv6 core protocol definition is in RFC 2460
RIPng, EIGRPv6, OSPFv3, ICMPv6, traceroute6, ping6, MP BGP-4 (mulitprotocol BGP v4)
ARP is replaced with NDP

Truncating addresses
4 digits informally a "quartet", so 4*4=16
0000:0000:0000:0000:0000:0000:0000:0000 = 128 bits

The Rules:
Leading 0's can be removed so 003A is 3A
Sets of consecutive 0's can be truncated to one in an address - 0000:0000 to 0:0
Replacing multiple quartets of 0's (0000:0000 with ::) can only be done once.
Doing the 3 in one address should be ok, within those rules
Expanding is easy: inverse the rules and pad out so there are 8 quartets of 4 bits each.

2000:1234:5678:9ABC::/64 prefix length is 64 bits (4 octets, 16 each)
In this way, a "/" prefix notation is almost iike a CIDR - it tells how many bits used.

Given Address	Expanded Form	Address's Prefix
34BA:B:B:0:5555:0:6060:707/80	34BA:000B:000B:0000:5555:0000:6060:0707	34BA:B:B:0:5555::/80
3124::DEAD:CAFE:FF:FE00:1/80	3124:0000:0000:DEAD:CAFE:00FF:FE00:0001	3124:0:0:DEAD:CAFE::/80
2BCD::FACE:BEFF:FEBE:CAFE/48	2BCD:0000:0000:0000:FACE:BEFF:FEBE:CAFE	2BCD::/48
3FED:F:E0:D00:FACE:BAFF:FE00:0/48	3FED:0000F:00E0:0D00:FACE:BAFF:FE00:0000	3FED:F:E0::/48
210F:A:B:C:CCCC:B0B0:9999:9009/40	210F:000A:000B:000C:CCCC:B0B0:9999:9009	210F:A::/40
34BA:B:B:0:5555:0:6060:707/36	34BA:000B:000B:0000:5555:0000:6060:0707	34BA:B::/36
3124::DEAD:CAFE:FF:FE00:1/60	3124:0000:0000:DEAD:CAFE:00FF:FE00:0001	3124:0:0:DEA0::/60
2BCD::FACE:1:BEFF:FEBE:CAFE/56	2BCD:0000:0000:FACE:0001:BEFF:FEBE:CAFE	2BCD:0:0:FA00::/56
3FED:F:E0:D000:FACE:BAFF:FE00:0/52	3FED:000F:00E0:D000:FACE:BAFF:FE00:0000	3FED:F:E0:D000::/52
3BED:800:0:40:FACE:BAFF:FE00:0/44	3BED:0800:0000:0040:FACE:BAFF:FE00:0000	3BED:800::/44

Global unicasts are initially delegated in blocks by IANA, ARIN etc., and are leased for usage by various service providers. Since there are so many IPv6 addresses, they exist as just a utility for network operation provisioning- there is no recognized range of addresses that is "owned" or has brand recognition (like Google's name server address of 8.8.8.8), you just get any block the size you need from an ISP and move it to another as needed.

2001:db8:2222::/48 -Company A
2001:db8:3333::/48 -Company B
And here are some global unicast versions of subnet and hostnames:
2001:db8:3333:0001::/64 -Company B's subnet 1
2001:db8:3333:0002::/64 -Company B's subnet 2
2001:db8:3333:2::1 -A host in Company B's subnet 2
2001:db8:3333:0003::/64 -Company B's subnet 3
2001:db8:3333:3::10 -A host in Company B's subnet 3
Initially started with 2 or 3; now anything not reserved

Global addresses have the same overall structure:
Global routing prefix (IANA- 48 bits) + subnet bits (16) + interfaceID (typically 64 bits) = 128 bits total
2001:0db8:1111 (48 bits): 0001(16 bits) :0000:0000:0000:0001 (64 bits)
In this example, subnets would be 0001, 0002, 0003

Unique Local Unicast Addresses are like private IPv4 addresses
FD (8 bits) - GlobalID (pseudorandom 40bits) - subnet 16 bits - remaining 64 bits for interface
You prefix with /48 and then add the subnet (64 total) and interface (the last 64 bits)
- FD00:1:1:0001::/64 would be a sample unique local

- FD + 00:1234:5678: + 9ABC:DEF1:2345:6789:ABCD

Static Unicast Address Configuration
Generating a Unique Interface ID using EUI-64 is usually done for us just by assigning the address with:
ipv6 address 2001:db8:1111:1::/64 eui-64

EUI-64 creates the interface ID part as follows:
Split the 6-byte (12-hex) MAC address in two halfs, and insert FFFE in between the two
Invert the seventh bit of the interface ID. Examples:

Interface MAC address is aa12:bcbc:1234
10101010 represents the first 8 bits of the MAC address (aa) - when inverting the 7th bit it becomes 10101000.
The answer is A8 and the EUI-64 address: 2001:0db8:0:1:a812:bcff:febc:1234 EUI-64

MAC address 0c0c:dede:1234
0c is 00001100 in the first 8 bits of the MAC address, which then becomes 00001110 when flipping the 7th bit.
The answer is then 0e and the EUI-64 address: 2001:0db8:0:1:0e0c:deff:fede:1234 EUI-64

MAC address 0b34:ba12:1234
0b in binary is 00001011, the first 8 bits of the MAC address, which then becomes 00001001.."
The answer is 09, and the IPv6 EUI-64 address: 2001:0db8:0:1:0934:baff:fe12:1234 EUI-64

```
interface GigabitEthernet0/0
 ipv6 address 2001:db8:1111:1::/64 eui-64
interface serial0/0/0
 ipv6 address 2001:db8:1111:2::/64 eui-64
show ipv6 interface brief
   GigabitEthernet0/0 [up/up]
      FE80::1FF:FE01:101                  <---link local, employing EUI64
      2001:DB8:1111:1:0:1FF:FE01:101   <---EUI64 global unicast address
   Serial0/0 [up/up]
      FE80::1FF:FE01:101
      2001:DB8:1111:2:0:1FF:FE01:101
```

Link-Local Addresses (LLA)
Auto-generated - interferfaces can create their own LLA
Common uses - Overhead protocols (NDP) and next-hop address
Forwarding scope is the local link only - (for within a subnet)
Unicast, represent a single host. Always starts with the same prefix: the first 10 bits are FE80::/10
64 Bits - FE80:0000:0000:0000 + 64 Bits - Interface ID: EUI-64
Second half can be formed with different rules
EUI-64 (like above)
Operating Systems use random processes
Manually configured

Key IPv6 Local-Scope Multicast Addresses

Short Name	Multicast Address	Meaning	IPv4 Equivalent
All-nodes	FF02::1	All nodes (using IPv6)	Subnet Broadcast Address
All-routers	FF02::2	All routers (using IPv6)	None
All-OSPF	FF02::5	All OSPF routers	224.0.0.5
All-OSPF-DR	FF02::6	All OSPF designated routers	224.0.0.6
EIGRPv6 routers	FF02::A	All routers (using EIGRPv6)	224.0.0.10

R1# **show ipv6 interface GigabitEthernet 0/0**
GigabitEthernet0/0 is up, line protocol is up
IPv6 is enabled, link-local address is FE80::1FF:FE01:101
[extra info omitted]

Joined group address(es):
FF02::1
FF02::2
FF02::A
FF02::1:FF01:101 <---------This host's solicited node multicast

Solicited-Node Multicast Addresses
A multicast address that is link-local in scope
Based on the unicast of the host, and only the last six hex digits
Each host listens for packets sent to **its** solicited-node address
Format: [Defined by RFC] + [Last 6 Hex Digits of Unicast]
Example: FF02:0000:0000:0000:0000:0001:FF_ _:_ _ _ _
By design, all hosts with the same last six hex digits use the same address
[FF02::1:FF/108 ?]

Special addresses used by routers:
:: (all 0's) signify unknown/ unspecified address Used when a host's own address is not yet known
::1 loopback - 127 binary 0s with a single 1

Neighbor Discovery Protocol (NDP):
Router discovery - available routers in the same subnet
- Router solicitation (RS) - Over FF02::1 (all routers MC), asks routers to "sound off"
- Router advertisements (RA) - Over FF02::1 (all nodes MC), router replies with link-local address, other info
Neighbor MAC discovery, NDP replaces ARP
- Neighbor Solicitation (NS, like an ARP request) and Neighbor Advertisement (NA, like an ARP reply)
- Duplicated Address Detection (DAD) uses NS requests and NA replies to keep track of unicasts
(SLAAC uses NS/NAs in DAD to do DHCP autoconfig)

Dynamic Unicast Address Configuration
Routers can be configured to use dynamic addresses
Two methods: interface FastEthernet0/0
ipv6 address dhcp <------Using Stateful DHCP
ipv6 address autoconfig <------- Using Stateless Address Autoconfiguration (SLAAC)

Regular DHCPv6
DHCP client sends out a solicitation using it's own link-local as the source address, and the "All-DHCP-Agents" MC - ff02::1:2) as it's destination

Configuring R1 to be a DHCP relay:
interface GigabitEthernet0/0
ipv6 dhcp relay destination 2001:db8:1111:3::8
R1# show ipv6 interface g0/0
GigabitEthernet0/0 is up, line protocol is up
Joined group address(es):
FF02::1
FF02::2
FF02::A
FF02::1:2 **<------ this got added for the relay role**
FF02::1:FF00:1

SLAAC: Using Stateless Address Autoconfiguration
Building an IPv6 Address Using SLAAC is easy:
A NDP RS is sent to get an RA to provide the local link prefix
The host then slaps on a EUI-64 or random value for an interface ID
Send out Neighbor Solicitation for DAD process to verify it's unique.

DHCPv6 vs SLAAC?
With stateless SLAAC, you get your DNS servers from the local router, and you don't need to lease an address. With a DHCP server being involved, you can have a lease and state information on a client. You might think DHCP would give more control, but it really doesn't since users can still use SLAAC instead.

CHECK THIS - question in practice set says SLAAC gets NDP to give it prefix length, default router addresses, but that Stateless DHCP provides the DNS servers.

Ping6 and Traceroute6, all the same
Extended is still just "ping" since it asks you the protocol

/128 is a "host route" for the router ip6 address

show ipv6 neighbors
show ipv6 routers -- to show connected routers

sh ipv6 interface GigabitEthernet 0/0

show ipv6 interface brief <--- no prefix length info
show ipv6 interface <--- Full interface details

ipv6 route ::/0
sh ipv6 neigh

Configuring static IP6 on two routers:

```
!R1
ipv6 unicast-routing              <---If not enabled, the router will act like a host and won't route
interface gigabitethernet0/0
  ipv6 address 2001:Ddb8:1111:1::1/64
interface serial0/0/0
  ipv6 address 2001:0db8:1111:0002:0000:0000:0000:0001/64
!R2
ipv6 unicast-routing
interface gigabitethernet0/0
  ipv6 address 2001:db8:1111:3::2/64
interface serial0/0/1
  ipv6 address 2001:db8:1111:2::2/64
```

#R1 **show ipv6 route connected**

```
IPv6 Routing Table - default 5 entries
!Omitted
C  2001:DB8:1111:1::/64 [0/0]
     via GigabitEthernet0/0, directly connected
C  2001:DB8:1111:2::/64 [0/0]
     via Serial0/0/0, directly connected
```

#R1 **show ipv6 interface GigabitEthernet 0/0**

```
GigabitEthernet0/0 is up, line protocol is up
```

```
  IPv6 is enabled, link-local address is FE80:1FF:FE01:101
  No virtual link-local address(es):
  Description: LAN at Site 1
  Global unicast address(es):
    2001:DB8:1111:1::1, subnet is 2001:DB8:1111:1::/64
#R1 show ipv6 interface s0/0/0
Serial0/0/0 is up, line protocol is up
  IPv6 is enabled, link-local address is FE80:1FF:FE01:101
  No virtual link-local address(es):
  Description: Link to R2
  Global unicast address(es):
    2001:DB8:1111:2::1, subnet is 2001:DB8:1111:2::/64
#R1 show ipv6 interface brief
GigabitEthernet0/0 [up/up]
    FE80:1FF:FE01:101
    2001:DB8:1111:1::1
GigabitEthernet0/1 [administratively down/down]
    unassigned
Serial0/0 [up/up]
    FE80:1FF:FE01:101
    2001:DB8:1111:2::1
```

http://freeccnalab.com/
http://www.packettracerlab.com/
http://www.packettracernetwork.com/labs/packettracerlabs.html
https://boubakr92.wordpress.com/2013/09/16/ccna-cheat-sheet-part-1/

ipv6 address <command>
ipv6 address
sh ipv6 route

- **Adding IPv6 Routes to Routing Tables**
  - Configuration of IPv6 addresses on working interfaces
  - Direct configuration of a static route
  - Configuration of a routing protocol on routers that share the same link
- **Rules for Connected and Local Routes**
  - Routers create routes based on each unicast address on an interface
    - Router creates a route for the subnet (a connected route)
    - Router creates a host route (/128) for the router IPv6 address
  - Routers do not create routes based on the LLA for the interface
  - Routers remove the connected and local routes if the interface fails
    - Re-adds when the interface is up/up

```
ipv6 unicast-routing
!
interface serial0/0/0
 ipv6 address 2001:db8:1111:4::1/64
!
interface serial0/0/1
 ipv6 address 2001:db8:1111:5::1/64
!
interface gigabitethernet0/0
 ipv6 address 2001:db8:1111:1::1/64
```

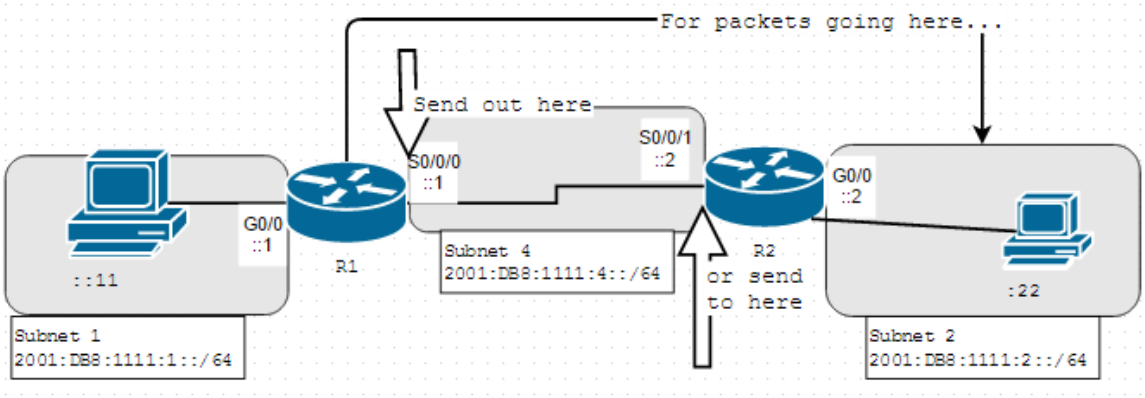
Before adding static or OSPF routes:

show ipv6 route

!Omitted
Codes: C - connected, L - local....
!Omitted
C 2001:db8:1111:1::/64 [0/0]
     via GigabitEthernet0/0, directly connected
!Omitted
C 2001:db8:1111:4::/64 [0/0]
     via Serial0/0/0, directly connected
!Omitted
C 2001:db8:1111:5::/64 [0/0]
     via Serial0/0/1, directly connected

show ipv6 route local

!Omitted
L 2001:db8:1111:1::1/128 [0/0]

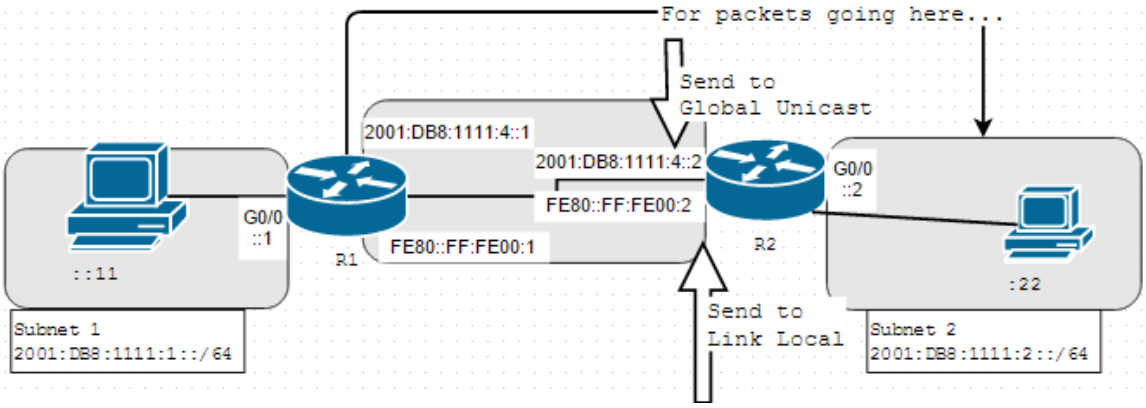


ADD

!Static route on router R1
R1(config)# **ipv6 route 2001:db8:1111:2::/64 s0/0/0**
!Static route on router R2
R2(config)# **ipv6 route 2001:db8:1111:1::/64 s0/0/1**

VERIFY

R1(config)# **show ipv6 route static**
S 2001:db8L1111L2LL/64 [1/0]
     via Serial0/0/0, directly connected
R1(config)# **show ipv6 route 2001:db8:1111:2::22**
Routing entry for 2001:db8:1111:2::/64
  Known via “static”, distance 1, metric 0
  Route count is 1/1, share count 0
  Routing paths:
    directly connected via Serial0/0/0
    Last updated 00:01:29 ago



Static Routes Using Next-Hop IPv6 Address: *Using Unicast or Link-Local as Next-Hop Address*

!First command is on R1, listing R2's global unicast
R1(config)#**ipv6 route 2001:db8:1111:2::/64 2001:db8:1111:4::2**
!This command is on R2, listed R1's global unicast
R2(config)#**ipv6 route 2001:db8:1111:1::/64 2001:db8:1111:4::1**
!Verify routes with **show ipv6 route static**

!First command is on R1, listing R2's link local address
R1(config)#**ipv6 route 2001:db8:1111:2::/64 s0/0/0 FE80:FF:FE00:2**
!This command is on R2, listed R1's link local address
R2(config)#**ipv6 route 2001:db8:1111:1::/64 s0/0/1 FE80:FF:FE00:1**
!Verify routes with **show ipv6 route static**

- **Static Default Routes**
  - Tells the router what to do when a packet does not match a route
    - Without a default route, router discards packet
    - With a default route, router forwards packet to the default route

!Forward out B1's S0/0/1 local interface...
B1(config)#**ipv6 route ::0 S0/0/1**
!Static Default Route
B1#**show ipv6 route static**
!Omitted
S ::/0 [1/0]

- Migration from OSPFv2 to OSPFv3
  - Before IPv6, the company supports IPv4 using OSPFv2
  - Company plans to use a dual-stack approach
  - Companies add OSPFv3 configuration to all the routers
    - OSPFv2 support stays in place

OSPFv2 Indirectly Enables OSPF on the Interface
and OSPFv3 Configuration Directly Enables OSPF

router ospf 1
 router-id 1.1.1.1
 network *10.0.0.0 0.255.255.255 area 0* *<--- indirectly enable on int*
interface s0/0/0
 ip address *10.1.1.1 255.255.255.0*

ipv6 router ospf 1

router-id 1.1.1.1
interface s0/0/0
ipv6 ospf 1 area 0 <----- directly enable on int

- **Configuring Single-Area OSPFv3**
  - OSPFv2:
    - If **router-id *rid*** OSPF subcommand is configured, use it
    - If the RID is not set, check the loopback interfaces with up status
    - Choose the highest numeric IP address
    - If neither are used, router picks highest numeric IPv4 address
  - OSPFv3:
    - Use **ipv6 router ospf *process-id***
    - Ensure the router has an OSPF router ID
      - Configuring the **router-id *id-value*** router subcommand
      - Preconfiguring an IPv4 on any loopback who is up
      - Preconfiguring an IPv4 on any working interface who is up
    - Configure the **ipv6 ospf *process-id* area *area-number*** on each OSPF interface

ipv6 unicast-routing
interface serial0/0/0
no ip address
ipv6 address 2001:db8:1111:4::1/64
!
interface s0/0/1
no ip address
ipv6 address 2001:db8:1111:5::1/64
!
interface GigabitEthernet0/0
no ip address
ipv6 address 2001:db8:1111:5::1/64
!Enabling OSPFv3 on three interfaces
config t
ipv6 router ospf 1
router-id 1.1.1.1
interface s0/0/0
ipv6 ospf 1 area 0
interface s0/0/1
ipv6 ospf 1 area 0
int gi0/0
ipv6 ospf 1 area 0
end

ipv6 unicast-routing
ipv6 router ospf 2
router-id 2.2.2.2
!
int s0/0/1
ipv6 address 2001:db8:1111:4::2
ipv6 ospf 2 area 0
!
int gi0/0
ipv6 address 2001:db8:1111:2::2
ipv6 ospf 2 area 0

To Display Details About...	OSPFv2	OSPFv3
OSPF process	show ip ospf	show ipv6 ospf
All sources of routing info	show ip protocols	show ipv6 protocols
Details about OSPF-enabled interfaces	show ip ospf interface	show ipv6 ospf interface
Concise info on OSPF-enabled interfaces	show ip ospf interface brief	show ipv6 ospf interface brief
List of neighbors	show ip ospf neighbor	show ipv6 ospf neighbor
Summary of LSDB	show ip ospf database	show ipv6 ospf database
OSPF-learned routes	show ip route ospf	show ipv6 route ospf

R1#**show ipv6 ospf**
Routing Process “ospfv3 1” with ID 1.1.1.1
!Omitted
Reference bandwidth unit is 100 mbps
Area BACKBONE 0
Number of interfaces in this area is 3
!Verifying interfaces
R1# **show ipv6 ospf interface brief**

Interface	PID	Area	Intf ID	Cost	State	Nbrs	F/C
Gi0/0	1	0	3	1	DR	0/0	
Se0/0/1	1	0	7	64	P2P	1/1	
Se0/0/0	1	0	6	64	P2P	1/1	

R1# **show ipv6 protocols**
IPv6 Routing Protocol is “connected”
IPv6 Routing Protocol is “ND”
IPv6 Routing Protocol is “ospf 1”
Interfaces (Area 0):
GigabitEthernet0/0
Serial0/0/1
Serial0/0/0

!First command is from R1, listing R2 and R2
show ipv6 ospf neighbor

Neighbor ID	Pri	State	Dead Time	Interface ID	Interface
3.3.3.3	0	FULL/ -	00:00:39	6	Serial0/0/1
2.2.2.2	0	FULL/ -	00:00:31	7	Serial0/0/0

!This command is from R2, listing R1 and R3
show ipv6 ospf neighbor

Neighbor ID	Pri	State	Dead Time	Interface ID	Interface
1.1.1.1	0	FULL/ -	00:00:39	6	Serial0/0/1
3.3.3.3	0	FULL/ -	00:00:31	7	Gi0/0

(Verifying OSPFv3 LSDB on R1)
show ipv6 ospf database
OSPFv3 Router with ID (2.2.2.2) (Process ID 2)
Router Link States (Area 0)

ADV Router	Age	Seq#	Fragment ID	Link count	Bits
1.1.1.1	452	0x80000002	0	2	None

2.2.2.2 456 0x80000004 0 2 None
3.3.3.3 457 0x80000005 0 2 None

(from R2)
show ipv6 route ospf
IPv6 Routing Table - default - 9 entries
!Omitted
O 2001:DB8:1111:1::/64 [110/65]
via FE80::FF:FE00:1, serial0/0/1

ipv6 unicast routing -- for static routing
:4::1/64
:5::1/64
:5::1/64
2001:db8:1111

show ipv6 route [local l static l]
show ipv6 route (address) to show info specific to that route
ipv6 route (address) s0/0 (next hop) --add to table - can be link local or global unicast of destination port

static default routes
ipv6 route ::0 s0/0/1
S ::0 will show up in routing table
(::0 any address)

same: router ospf 1, router-id 1.1.1.1 - you should consider rid to be required in ipv6
OSPF instead of indirectly enabling on interfaces ipv6 has to have "ipv6 ospf 1 area 0"
(no "network ip mask area command for router part)

ch 27-29

Address Description Available Scopes
ff0X::1 All nodes address, identify the group of all IPv6 nodes Available in scope 1 (interface-local) and 2 (link-local):

ff01::1 → All nodes in the interface-local
ff02::1 → All nodes in the link-local

ff0X::2 All routers Available in scope 1 (interface-local), 2 (link-local) and 5 (site-local):

ff01::2 → All routers in the interface-local
ff02::2 → All routers in the link-local
ff05::2 → All routers in the site-local

ff02::5	OSPFIGP	2 (link-local)
ff02::6	OSPFIGP Designated Routers	2 (link-local)
ff02::9	RIP Routers	2 (link-local)

ff02::a	EIGRP Routers	2 (link-local)
ff02::d	All PIM Routers	2 (link-local)
ff02::1a	All RPL Routers	2 (link-local)
ff0X::fb	mDNSv6	Available in all scopes
ff0X::101	All Network Time Protocol (NTP) servers	Available in all scopes
ff02::1:1	Link Name	2 (link-local)
ff02::1:2	All-dhcp-agents	2 (link-local)
ff02::1:3	Link-local Multicast Name Resolution	2 (link-local)
ff05::1:3	All-dhcp-servers	5 (site-local)
ff02::1:ff00:0/104	Solicited-node multicast address. See below	2 (link-local)
ff02::2:ff00:0/104	Node Information Queries	2 (link-local)

NAT*: s=172.16.2.2, d=192.168.2.1->10.1.1.1 [1]

Notice the last line's asterisk (*). This means the packet was translated and fast-switched to the destination. ***Fast-switching aka cache-based switching aka "route one switch many"*** - this process is used on Cisco routers to create a cache of layer 3 routing information to be accessed at layer 2 so packets can be forwarded quickly through a router without the routing table having to be parsed for every packet. As packets are packet switched (looked up in the routing table), this information is stored in the cache for later use if needed for faster routing processing. Note the count in brackets- it shows that after 3 or 4 exchanges, fast switching was turned on for this pair of hosts.

Before a host can send ICMP (ping) packets to another device, it needs to learn the MAC address of the destination device so it first sends out an ARP Request. In fact, the first ping packet is dropped because the router cannot create a complete packet without learning the destination MAC address.

no ip http server -- Shuts off the Cisco http interface (if available)
no service tcp-small-servers - and - no service udp-small-servers
These commands disabled these services: echo (7), discard (9), daytime (13), chargen (19)
These are disabled by default in Cisco IOS 12 and later.
See [http://www.cisco.com/c/en/us/support/docs/ip/access-lists/13608-21.html] for more info on shutting down services.