CCIE Notes

ATM

If you are having trouble with ATM, enable ilmi (`atm pvc 2 0 16 ilmi`) and do a show atm ilmi-status. This will show if you are communicating with the switch.

ATM PVCs

For ATM pvc's, you can either use maps (similar to Frame Relay maps) or inarp. Inarp will only work with IP, so if IPX is also involved you must use maps (this may vary with IOS version). Inarp is off by default on a pvc. Enable it simply by including the inarp keyword in your `atm pvc` command. If you do not include it, you must use the `map-group` and `map-list` commands to manually define mappings.

ATM SVCs

ATM SVCs are still fair game, even without LANE. For this method you define a pvc for the signaling protocol, qsaal (`atm pvc 1 0 5 qsaal`), and optionally one for ilmi (`atm pvc 2 0 16 ilmi`). However in this case you have two choices:

You can map (using `map-group` and `map-list`) IP or IPX addresses to full, 20-byte ATM addresses. The router then uses qsaal to signal for the ATM switch to construct an SVC to the ATM address in the map statements. This is obviously clumsy.

The other alternative is to use ATM ARP Server (IP only?). With this, set the server using the `arp server self` command. Then on each client define the server's 20-byte ATM NSAP using the `atm arp server` *`address`* command.

ATM PVC Discovery

This method only uses one PVC – ilmi (`atm pvc 2 0 16 ilmi`) to discover VC's. Use the `atm ilmi-pvc-discovery` command on the main ATM interface. This will let the switch announce PVC's. This also performs ATM mapping for network layer addresses. This does not require qsaal (`atm pvc 1 0 5 qsaal`). It does "stick" them on the main interface – so if you don't want them there, write down the VPI/VCI's, turn off discovery and configure the PVC(s) on your subinterface. Another alternative is to use the `atm ilmi-pvc-discovery subinterface` command. This places the PVC in the subinterface with the same number as the VPI of the PVC.

BGP

Filtering

To filter routes you can use a `neighbor dist-list`, just `dist-list` or a `neighbor route-map` with only a `match ip address` statement. Using just a `dist-list` filters them from the routing table but leaves them in the bgp table. The other two eliminate them from both. An extended access list like

`access-list 102 permit ip host 10.10.10.0 host 255.255.255.0`
seems to work with the first and last option, but not the “plain” dist-list option…

When filtering based on AS path, using ^ (to denote the beginning of an AS path) matches the beginning of the path as it is listed in the bgp table. For example, to match:

	Network	Next Hop	Metric	LocPrf	Weight	Path
*	i3.0.0.0	137.39.23.89	1000	50	0	701 80 i

You could use:
sho ip bgp reg ^701_80_

Even though the true “beginning” of the AS path is 80 (that is, the route was originated from AS 80).

Communities

In order to send communities, you need to enter the `neighbor 10.13.13.1 send-community` command. This will send any communities the BGP routes already have to that neighbor. Communities are not sent by default – they need this command!!!

In order to tag routes with communities, you need:

```
neighbor 192.168.1.2 send-community
neighbor 192.168.1.2 route-map setcommunity out
route-map setcommunity permit 10
 match ip address 2
 set community no-export
!
route-map setcommunity permit 20
!
access-list 2 permit 192.168.254.0
```

You need the second route-map statement to send “all other” routes without communities. Also, it is helpful to use the global command `ip bgp new-format`. Otherwise your communities look really weird!

Synchronization

Turn off whenever possible! With it on, all iBGP learned routes must also show up in some IGP (OSPF,etc.) Even static routes are not enough!

Aggregate Address

This is a useful command for summarizing an address block. Use the keyword `summary-only` to suppress more specific routes. However to advertise a summary at least one more specific route must be in the router’s BGP table (via a network command, redistribution, etc.)

The `summary-only` keyword only appears to suppress more specific routes that are within the natural class defined by the aggregate address and mask. That is, you can specify an address/mask that is larger than its natural mask. The exact address/mask you specified will get propagated via BGP, however it will only suppress more specific routes within its own natural address class.

Bridging

For bridging over Frame-Relay, there are no special requirements if all interfaces are point-to-point. However for Frame Relay (or ATM) physical or multipoint interfaces, you need one `frame-relay map bridge` *`dlci`* `broadcast` command for each DLCI that's part of physical or multipoint interfaces. However, note that for physical and multipoint interfaces, the router will not forward packets out the same physical or multipoint interface that bridge packets were received on (regardless of all else, including Spanning Tree)!

Spanning Tree

The root bridge is determined by the lowest bridge priority – set by the global `bridge priority` command.

On each subnet a designated bridge is elected. This is the bridge that will have the forwarding path to the root. The bridge with the lowest cost path to the root will be the designated bridge (and thus will be forwarding). In the case where two or more bridges have the same path cost to the root, the bridge with the lowest priority becomes the designated bridge.

The path cost is calculated by adding the "outbound" path costs of all paths to the root. That is, path costs are added as you are leaving each router on the way to the root (the path cost as you enter a router is irrelevant).

All non-root bridges will have exactly one root port. These listen for BPDUs from the root bridge. Non-root bridges will send BPDUs out all their designated ports. For all non-root bridges, if a port is not a root port and not a designated port, it is a blocked port.

Port priority is almost never used. The only time this might be used is if two non-root bridges had redundant links between them. One of the four ports for those two links would have to block – port priority would allow you to control which one it was. If you don't set this on any of the four, the IOS will select one to block (but how? Who cares?).

IRB/CRB

With CRB for a given protocol (IP or IPX), there will be a group of routed interfaces and a group of bridged interfaces. The routed interfaces each get an IP (and IPX) address and can route to any other routed interface – but not to the group of bridged interfaces. The bridged interfaces can bridge between each other, but not route to the routed interfaces (the bridged interfaces don't even get an IP or IPX address). CRB is not terribly useful.

With IRB you may have the same set of routed and/or bridged interfaces, but you can easily establish connectivity between them.

When you configure IRB or CRB you have four choices for each protocol:

1. `bridge 1 route ip`
   `bridge 1 bridge ip`
   Use this to bridge the protocol among interfaces within the bridge group, but route it to all other interfaces. (Very common). For interfaces within the IRB bridge-group 1, configure the protocol information on `int bvi1`, not on the "real" interfaces.
2. `no bridge 1 route ip`
   `bridge 1 bridge ip`
   Use this to bridge the protocol among interfaces within the bridge group, but not route it to any interfaces outside of the bridge group. Do not configure protocol information on `int bvi1` or on the "real" interfaces within the bridge group.
3. `bridge 1 route ip`
   `no bridge 1 bridge ip`
   Use this to route the protocol among all interfaces – within the bridge group and outside the bridge group. Configure the protocol information on all the "real" interfaces (within and outside the bridge group) but not on `int bvi1`. This is common when you want to route one protocol (like IP) but bridge another (like IPX).
4. `no bridge 1 route ip`
   `no bridge 1 bridge ip`
   You would probably never use this. This would 'turn off' the protocol for the entire bridge group – you would not bridge it between interfaces in the bridge group, nor would you route it to any interfaces outside the bridge group.

Debug

If you need to use `debug ip packet [detail] [`*`access-list`*`]`, remember that only packets that are processed switched will get debugged. To disable fast switching (and force process switching) use `no ip route-cache` on each interface (especially the incoming interface for the packets in question).

Dial

My dial strategy is going to be to use the simplest (most dependable) solution unless directed otherwise. My order of preference for IP will be:

1. Floating Static Routes
2. IP OSPF Demand Circuit
3. Dialer Watch
4. Snapshot routing
5. Dial Backup

My order of preference for IPX will be:

1. Floating Static Routes
2. Tunnel IPX through IP (especially effective if using 1, 2 or 3 above)
3. Snapshot routing
4. Dial Backup

The 2503's and 2504's typically have an S/T ISDN interface. A 2524 often will have a U.

Floating Static Routes

For IPX to use a static, default route, the WAN (i.e., ISDN) must use IPXWAN! IPXWAN needs an internal-network number first!

SnapShot Routing

Remember, snapshot routing only works with RIP (IP), IGRP (IP), RIP and SAP (IPX).
Even with Snapshot routing you still need the same old dialer map statements that you always have (typically)…plus one or more for snapshot.

PPP Authentication

You want to indicate `ppp authentication chap` under the physical interface (dialer maps) or the physical and logical interface (dialer profiles). If you don't want one side to use chap (if you don't want that router to challenge the other) omit the `ppp authentication chap`. However if the opposite router has ppp authentication chap, you must have the other router's name & password in your database.

For PAP authentication, you need the same config as with CHAP, yet also the receiving router seems to also need a `ppp pap username r4 password 0 cisco`, where r4 is that router's own hostname and cisco is the password.

Distribute Lists

* Try adding the word log at the end of an access-list statement to log what is happening with the access list.

Distribute lists “in” block routes from the routing table, but not the (OSPF or other) database. This will block the routes from appearing in that router, but not in other routers that run (OSPF or other) and get the same Link State Database.

Distribute lists “out” are typically much more effective from blocking a route from a large portion of the network. However with OSPF `distribute-list out` only works on External Type 1 or 2 routes – not with internal OSPF routes.

Distribution lists may not take effect immediately. You may have to bounce the interface or do a `clear ip route *` to activate them.

The `distribute-list` *`list#`* `out` *`process`* is very tricky. For example:

```
2501b(config)# router ospf 103
2501b(config-router)#distribute-list 16 out eigrp 1
```

It would appear that this would regulate what ospf sends out to eigrp 1. But instead it controls what OSPF receives in from EIGRP 1 (or, more aptly, what EIGRP sends out to OSPF).

DLSw

Here is a brief overview of the types of DLSw transports:

DLSw also uses noncanonical (T.R.) format for mac addresses.

DLSw will automatically convert between Ethernet and Token Ring stations if they are located on different routers. In order to get Ethernet and Token Ring stations to communicate on the same router, SR-Translational bridging must be enabled.

TCP – probably the most robust DLSw implementation – recommended.
FST – does not perform local acknowledgement, supports Token Ring only, fewer queuing options.
Direct – supports HDLC and Frame-Relay only, fewer queuing options (No IP encapsulation).
LLC2 (lite) – less overhead but also less rerouting, Frame-Relay only.

DLSw chooses 1 path by default, but can be configured to use multiple paths.

DLSw can choose paths based on cost. Cost in a local-peer statement is what is advertised out to all remote peers. Cost in a remote-peer statement sets the cost to connect to that peer.

DLSw can limit the MTU size (handy going from TR to Eth) using the `lf 1500` keyword and value on the `remote-peer` statement.

Filtering

With `dlsw prom-peer-defaults` and `dlsw peer-on-demand-defaults` all filters (dmac-output-list, host-netbios-out, lsap-output-list, etc.) are *outbound* to other peers (not outbound to the LAN interface).

With `dlsw remote-peer` statements all filters (dmac-output-list, host-netbios-out, lsap-output-list, etc.) are *outbound* to other peers (not outbound to the LAN interface).

A local DLSw peer can specify `dlsw remote-peer 1 tcp 10.10.10.10`. This command refers to list 1. It can be port list 1, ring list 1 and/or bgroup list 1. This command limits what the remote peer (in this case 10.10.10.10) can access locally (on the peer on which it is defined).

Border Peers/Peer Groups

By default for DLSw to have "full mesh" connectivity, you need a full mesh of DLSw connections. The exception is peer groups. With peer groups you can group DLSw routers into groups. Within a group each router only needs a connection to the bordrer peer(s). The border peer forwards broadcasts to all other peers within the group as well as any other border peers (from different groups) that are configured (basically acting like a BGP route reflector). Once the explorer finds its destination, a connection is setup router ←→ router (listed in the routers as peer-on-demand, or simply **pod**), even if the routers are in different groups.

Usually in this case use promiscuous peering. That is, all routers will likely need to be configured to accept any connection (promiscuous) since they could be getting connections from many routers.

Note: in the above scenario you will get promiscuous peers and pod (peer on demand) peers. To filter these use `dlsw prom-peer-defaults` and `dlsw peer-on-demand-defaults` to filter! Remember – these filters are **outbound** to other peers!

TCP connections

DLSw sets up connection on TCP ports 2065 and 2067. DLSw allows for a TCP connection to be built using one of these ports (likely 2065) in each direction. However if the DLSw routers can accommodate only one bi-directional connection (this will almost always be the case for Cisco routers), one TCP connection gets torn down. The router with the higher DLSw peer IP Address tears down the connection. Watch this if you have to NAT a DLSw peer address! Also its best to allow TCP 2065/2067 both ways through an access-list, even if the "steady state" DLSw coinnection will only require it in one direction.

EIGRP

If you have to run EIGRP over a dial interface, I recommend using `dialer watch-group`.

For NBMA topologies (Frame-Relay, ATM) EIGRP can have split-horizon disabled for spoke-spoke reachability (true for both IP and IPX).

Frame Relay

If you see a PVC with the status of "deleted," it probably means you typed in an `interface-dlci 100` command, but the frame switch is not announcing (and doesn't know about) that DLCI – check DLCI.

If you see a PVC with the status of "inactive," it probably means the local router's connection to the frame switch is fine, but there is a problem with the 'far' end of the PVC. Check the router that is supposed to terminate the PVC.

If you use a `frame-relay map` statements, you don't need `frame-relay interface-dlci` command(s) (unless you need to do traffic shaping). It may be a good idea to only use the map statements.

In Frame Relay you may want to place a map statement for your own IP address so that you can ping it (or ask the proctor if this is necessary).

Inverse Arp and Mapping

Frame Relay needs a way to connect, or map, a Layer 3 address (IP or IPX address) with a particular Frame Relay DLCI. That is, when a router attempts to forward packets to an IP or IPX address it needs to know out which virtual circuit – specified by a Frame Relay DLCI – the packet should be forwarded.

In some cases (such as where two routers are connected by a single virtual circuit, i.e., a single DLCI) the routers can use inverse-arp to determine the Layer 3 (IP or IPX) address at the opposite end of the virtual circuit. However in other cases, such as two "spoke" Frame Relay sites connected by one "hub" Frame Relay site, the two spoke can not use inverse-arp to learn each other's Layer 3 addresses. This is because inverse-arp packets are never forwarded (in this example, they are not forwarded by the "hub" router).

In these cases it is common to manually map (define) each Layer 3 address the router can reach to a specific DLCI (virtual circuit). Using sub-interfaces is an easy way to avoid doing this, but when does the CCIE exam ever take the easy way?

Also, if you perform mapping on a router, it is best to map every router, including the hub router. Even if connectivity exists between that router and the hub router, if you are mapping other remotes make a habit of mapping the hub router as well. In some version of IOS inverse-arp is disabled once a Frame Relay mapping occurs, however the problem this poses is often not apparent until the next reboot.

The way this can occur is as follows: suppose router A is a “spoke” router connecting to router B. Router C is also a spoke router that connects to router B. Router A uses inverse-arp to map router B’s IP address to a particular DLCI. However router A can not inverse-arp for router C’s IP address as discussed. A map statement is placed in router A for router C. Everything works great since you router A has the two mappings it needs: a dynamically learned one for router B (via inverse-arp) and a manually learned one (via a map statement) for router C.

However with some versions of code the map statement disables inverse-arp. Thus once the router is rebooted is loses its dynamically learned mapping for router B. Since the map statement has disabled inverse-arp, connectivity is lost. Thus, to be safe if you are performing map statements. add one for each router in the Frame cloud.

Central Site Frame Relay router		Remote Site Frame Relay router	
Interface	**Issues**	**Interface**	**Issues**
No subinterfaces	May need to disable IP/IPX split horizon.	No subinterfaces	Need a frame-relay map statement for **all** neighbors. Need ip ospf priority 0 on all remotes. Need to enable IP, IPX split horizon.
No subinterfaces	OSPF network type mismatch – probably have to use ip ospf network point-to-multipoint to make it work. May need to disable IP/IPX split horizon.	Point-Point subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch – probably have to use ip ospf network point-to-multipoint to make it work.
No subinterfaces	Very unlikely configuration. May need to disable IP/IPX split horizon.	Multipoint subinterfaces	Need frame-relay interface-dlci command. Need either: • On remotes: a frame-relay map statement for **all** neighbors and ip ospf priority 0, or • ip ospf network point-to-multipoint everywhere.
Point-Point subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch.	No subinterfaces	OSPF network type mismatch – set remotes to ip ospf network point-to-point. Remotes will be on different subnets. Need to enable IP, IPX split horizon.
Point-Point subinterfaces	Need frame-relay interface-dlci command.	Point-Point subinterfaces	Need frame-relay interface-dlci command.
Point-Point subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch. Very unlikely configuration.	Multipoint subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch – set remotes to ip ospf network point-to-point.
Multipoint subinterfaces	Need frame-relay interface-dlci command. Need to disable IP, IPX split horizon.	No subinterfaces	Need a frame-relay map statement for **all** neighbors. Need ip ospf priority 0 on all remotes. On 11.3 and lower, need ip ospf network point-to-multipoint or statically defined OSPF neighbors. Need to enable IP, IPX split horizon.
Multipoint subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch – probably have to use ip ospf network point-to-multipoint to make it work. Need to disable IP, IPX split horizon.	Point-Point subinterfaces	Need frame-relay interface-dlci command. OSPF network type mismatch – probably have to use ip ospf network point-to-multipoint to make it work.
Multipoint subinterfaces	Need frame-relay interface-dlci command. Need to disable IP, IPX split horizon. Very unlikely configuration.	Multipoint subinterfaces	Need frame-relay interface-dlci command. Need either: • On remotes: a frame-relay map statement for **all** neighbors and ip ospf priority 0, or • ip ospf network point-to-multipoint everywhere.

When configuring your `frame-relay map` statements, don't forget the `broadcast` at the end! For bridging, have the "hub" frame relay router be the root of the spanning tree! For ISIS, add a `frame-relay map clns` *`dlci`* `broadcast` command!

OSPF

A Frame Relay interface (not a subinterface) defaults to OSPF network type of nonbroadcast (NBMA). If using the default non-broadcast network type, be sure to set `ip ospf priority 0` on all remotes.

A Frame Relay point-to-point subinterface defaults to OSPF network type of point to point.

A Frame Relay multipoint subinterface defaults to OSPF network type of nonbroadcast (NBMA).

If you use point-to-point subinterfaces at one end of a PVC and no subinterfaces at the end, you must account for the type mismatch. For example, use `ip ospf network point-to-point` at the end not using subinterfaces. If you use a combination of physical and multipoint subinterfaces, use `ip ospf network point-to-multipoint`.

If you can't use broadcasts (as with the frame relay map statements or if you must use `ip ospf network point-to-multipoint non-broadcast`, for example) you must manually define OSPF neighbors with the neighbor statement.

Getting Started Checklist

It is easy to gather enough information about the lab to be able to prepare a "getting started" checklist. This is a list of the first steps to take on the morning of the first day of the lab. Here is my list, in order:

1. Read the lab exam twice. Yes, **twice.** Don't skim it and read it – read it twice. Make a list of:
    a. Hidden issues and pitfalls
    b. Your strong and weak areas
2. In between the first and second readings, configure the terminal server to connect to every router and switch in your rack. Make r1 the first connection, r2 the second connection, etc. Make the switches the last connections.
3. Check and record the IOS version, IOS image (name – feature set) and interfaces on each router.
4. Unless they are in the initial configuration script, write erase & reload each router. This will assure a clean start. It will also verify that they will reload properly – you don't want to discover they have a problem rebooting at 3:30! If you do this in between readings, the routers will have plenty of time to reload.
5. Create an IP address matrix. Don't waste time making something that can be hung at the Museum of Fine Arts when you're through. Just make a very simple line for each major network. Create major

divisions – 64, 128, 192, etc. Write down each address that is required and each one that you select.

6. Begin cabling the routers per the lab.
7. In notepad enter all commands that will be entered into *every* router.
8. Configure the routers with the above commands as well as all "layer 2" commands. Verify:
    a. Show isdn status yields "`MULTIPLE_FRAME_ESTABLISHED`"
    b. Verify all interfaces are up, up
    c. Verify all Frame-Relay PVCs are `LOCAL` and `ACTIVE`

IGRP

When changing the distance on IGRP you should perform a `clear ip route`. Otherwise IGRP seems to wait until the routes clear naturally (holddown, etc.) before they are reinstalled with the new distance.

IGRP does not exchange the default route (0.0.0.0) as most of the other protocols do. If you must have a default route with a router running IGRP, use:

```
ip default-network 172.16.0.0
```

where 172.16.0.0 is a classful network. This classful network (not a subnet) must be in your routing table. It also must be part of the IGRP process (either via the network statement or redistribution). This will cause this router to **generate** a candidate default route via IGRP.

Remember, IGRP has a lower admin distance than OSPF. That's probably not what you want!!

IKE

IKE is the Internet Key Exchange standard and is usually performed using the ISAKMP protocol. IKE is often used with IPSec because it automates key management and controls the security associations that are formed, though IKE is not required for IPSec. IKE policies define five things:

- encryption algorithm (such as **des**)
- hash algorithm (such as **sha** or **md5**)
- authentication method (such as **rsa-sig**, **rsa-encr** or **pre-share**)
- Diffe-Hellman group (such as **group-1** (768-bit) or **group-2** (1024 bit))
- security association lifetime (in seconds)

All of these have defaults (and the defaults can be used) except authentication – that must be specified. Pre-share is by far the easiest authentication method. Rsa-sig authentication requires a certificate authority (and thus is very unlikely to be on the CCIE Lab). These parameters affect the data that flows between hosts during the IKE

negotiation – not the actual data flows. Encryption and authentication of data flows is defined by the transform set in IPSec.

IPSec

To configure IPSec:

- Determine whether to use ISAKMP (recommended) or manual config for security associations
- Configure ISAKMP (recommended) or manual IPSec

Then:

1. Define the ISAKMP policy (ISAKMP only)
2. Define the keys (pre-shared, RSA, etc.)
3. Define a transform set (security configuration)
4. Define an access list to determine what traffic will be sent via IPSec
5. Create crypto map entries
6. Apply the crypto map to an interface

It appears IPSec likes to have the crypto map applied to the "outer most" interface. In the past I have applied the `crypto map` statement to the LAN (inside) interfaces and had no success (even if the routers are IPSec peering between loopbacks).

Access lists

For ipsec-manual mode (not using IKE/ISAKMP), only 1 access list entry is permitted; all others are ignored.

Always make access lists mirror images of each other at opposite ends!

Don't use the `any` keyword in your access lists.

IPSec through a Tunnel Interface

For running IPSec through a tunnel, first define the tunnel between the two physical interfaces on each router. Once the tunnel is working, define the IPSec peers between loopback interfaces. To do this you will need the `crypto map mymap local-address loopback 0` command (to set the peer's local IPSec peer address).

You will need some routing so that each router knows of the other's loopback address – static routing, a routing protocol through the tunnel, etc.

Enable the crypto map on both the physical interface and the tunnel interface.

IPX

When enabling IPX routing globally, use `ipx routing x.x.x` where x=router number. Thus, router 1 would be `ipx routing 1.1.1` (easier for access lists, IPX pings, etc.)

When you apply `ipx routing 1.1.1` to a router, it applies that "host" address to all non-LAN interfaces (serial, ISDN and loopback interfaces). The internal network always uses 0000.0000.0001, and the LANs use the "natural" mac address.

An IPX router will not install a SAP for which the network for that SAP was learned upon a different interface. Watch this on distribute lists.

When you enable ipx router eigrp, rip will stay on for those eigrp interfaces until you turn it off with the `no network ###` under `ipx router rip`. When nlsp is enabled on an interface, rip packets will only be sent if the other side is also sending rip packets – which is unlikely. RIP and SAP may be manually turned off with the `no ipx nlsp rip` and `no ipx nlsp sap` commands.

Filtering

Use distribute lists on a routing process – rip, eigrp or nlsp. If you do not specify an interface for the distribute list, the list gets applied to all interfaces.

For:

```
ipx router rip
 distribute-list 800 out eigrp 1
```

It would appear that this would regulate what RIP sends out to EIGRP 1. But instead it controls what RIP receives in from EIGRP 1 (or, more aptly, what EIGRP 1 sends out to RIP). [Acts same as IP]

For SAP filtering, use `distribute-sap-list` under the appropriate ipx router command. This lets you specify the source network (with mask), any source network, source network and host, service type (i.e. 4=file server), server name. Acts just like the distribute-list for routing updates.

The `ipx router-filter` interface command allows you to specify routers from which you will or will not accept updates – however it only works for RIP and EIGRP (not NLSP). With this command you can specify exact N.H.H.H routers or just permit/deny any router from a given network by only listing the network.

You can filter input (received updates) or output (sent updates) networks or SAPs on an interface level with one of the following commands:

```
ipx input-network-filter
ipx input-sap-filter
```

```
ipx output-network-filter
ipx output-sap-filter
```

Useful things to know for extended filters:

IPX Protocol (also called “packet type”) is similar to IP protocol (TCP, UDP, GRE, etc.) Usually set this to “any” in filters.

IPX RIP uses socket number 453.
IPX SAP uses number 452.

File servers use server-type 4 (in SAPs)
Print servers use server-type 7 (in SAPs)

EIGRP

Split Horizon may be disabled when using IPX EIGRP with the `no ipx split-horizon eigrp 1` command. This is handy on the hub router of Frame-Relay or ATM networks.

To use just EIGRP (and not RIP/SAP) on a WAN link, use the `network xxx` command under `IPX router eigrp 1` and the `no network xxx` under the `IPX router rip`. On WANs this will enable SAPs automatically. On LANs this will enable routing updates, but not SAPs. To enable EIGRP to propagate SAPs on LANs, use `ipx sap-incremental eigrp 1`.

Redistribution

RIP ←→ NLSP automatic
RIP ←→ EIGRP automatic
EIGRP ←→ NLSP not automatic, must be specified
Connected → EIGRP automatic (!!)
In any case it can be controlled via an access list. When using an ACL to control redistribution, deny means deny explicit routes – only aggregates!
Permit means permit explicit routes.
NLSP ←→ NLSP (with different area tags):

- Not automatic, use “route-aggregation” command
- When aggregation is turned on, its by area address
- It seems to advertise the area summary, but also explicit routes. To prevent explicit, use the `redistribute access-list` command.

NLSP

When configuring NLSP with multiple areas, use a tag, such as area1 or area2. Use it both in the global `ipx router nlsp area1` command as well as the interface `ipx nlsp area1 enable` command.

ISIS

Two ISIS nodes with the same area address will exist in the same area. This will allow them to establish a Level 1 relationship. ISIS with different area addresses will exist in different areas and establish a Level 2 relationship. By default ISIS routers can be level 1/2 routers, though this can be manually defined globally (`is-type`) or on each circuit (`isis circuit-type`).

It appears only the first 4 characters in an ISIS address denote the area. For example, the address 49.2222.0000.0000.5555 should have an area address of 49.22. However for simplicity make all 4 digits in the first group of four the same (and equal to the area address – 49.2222, for example). Make the last four characters (5555, for example) all the same (as opposed to "0005"). This represents 'router5' or 'r5'.

ISIS uses two different network types and they cannot be changed. Frame-Relay physical and multipoint interfaces are one type; point-to-point interfaces are the other type. These must match for adjacencies to form. If no adjacencies are forming use the `debug isis adj` command. The only other alternative is to create a tunnel.

An ISIS interface does not get its own address. It uses the NET of the router to which it connects. However it does get its own circuit ID. The circuit ID is the router's NET with the selector byte (last byte) incremented.

show clns neighbor
show clns interface
debug isis adj

Multicast

IGMP/CGMP

IGMP (Internet Group Management Protocol) is the standard multicast protocol that controls hosts joining multicast groups (and thus determines where a router needs to forward multicast traffic). CGMP is Cisco's proprietary IGMP. It only goes between the router and the switch, telling the switch on what ports it needs to forward multicast traffic.

PIM

PIM (Protocol Independent Multicast) is one of the leading multicast standards. PIM can operate in sparse mode, dense mode or sparse-dense mode.

In dense mode, multicast routers assume all other multicast routers want multicast flows. If a multicast router has no clients for a flow, it can send a Prune message back toward the source to stop that flow.

In sparse mode, multicast routers assume all other multicast routers do not want multicast flows. A multicast router must specifically request a flow (based on the requests of its clients). Mutlicast routers and multicast sources are tracked by a rendezvous point (RP).

In sparse-dense mode the interface acts either like sparse mode or dense mode for each multicast group. Thus an interface can be in sparse mode for some groups and dense mode for other groups.

DVMRP

DVMRP (Distance Vector Multicast Routing Protocol) is not fully supported by Cisco. However Cisco can send and receive packets from a DVMRP router.

NTP

For basic NTP configs, see Rob's Study Sheet. On the "master" or "server" router to control what routers can access NTP:
ntp access-group serve 71
Where 71 is the ACL that restricts what routers can use NTP and where serve is…
To use authentication with NTP, use:

OSPF

If you have a partial mesh Frame-Relay network (likely) and you are forced to use the non-broadcast network type (as opposed to the more favorable point-to-multipoint type) you will likely have to manually configure neighbors. In this case you will probably only need to define these at the hub router. Use `ip ospf priority 0` at the remotes.

You don't need `no auto-summary`.

Loopback networks won't be part of the OSPF process by default – they need to be added with the `network` statement. Loopback networks get defined as host routes (/32 mask) regardless of the "real" mask. However if you want the "whole loopback subnet" to be visible to the rest of the network, consider:

- Placing the loopback in its own area and summarizing:

```
interface loopback0
  ip address 192.168.253.1 255.255.255.0

router ospf 1
  network 192.168.253.0 0.0.0.255 area 4
  area 4 range 192.168.253.0 255.255.255.0
```

- Defining the ospf network type as point-to-point (V12.0 and later)

```
interface loopback0
  ip address 192.168.253.1 255.255.255.0
  ip ospf network point-to-point
```

Network Types

In Frame Relay you typically need `frame-relay map` statements if using no subinterfaces (just the physical interface) or point-to-multipoint subinterfaces on the remotes (in a partial mesh topology).

This is even true if you use the default OSPF network types for these interfaces, non-broadcast (NBMA). However if you use `ip ospf network point-to-multipoint`, you do not need `frame-relay map` statements (at least for IP) – even at the remotes. This is because this network type makes the Frame Relay partial mesh look like a collection of point-to-point networks.

In a hub-and-spoke frame relay network, if the hub is using a point-to-multipoint subinterface, either `ip ospf network point-to-multipoint` is required or manually defining neighbors at the hub is required. This is true in 11.3, though it seems to have been removed in 12.0.

Distance

You can set distances for intra-area, inter-area and external OSPF routes with the `distance ospf` router command. This can control what routes the router chooses to place in the routing table.

Summarization

When you use an `area 1 range` command it will summarize all OSPF internal routes, but none of the OSPF external (type 1 or 2) routes.

When you use the `summary-address 172.17.0.0 255.255.0.0` command, it does the opposite: it summarizes all OSPF external routes but none of the internal OSPF routes. It also only works on routers that are the ASBR for the external routes being summarized. Also, the summary advertisement seems to be an external type 2 route, with the metric being the lowest of the routes within that range.

However the OSPF `summary-address` command can also summarize external (type 1 or type 2) OSPF routes that are being redistributed into another protocol from OSPF. This can be very useful for IGRP and RIP, which are bound by FLSM. For example, OSPF can use the `summary-address` command to summarize many /27 OSPF networks into a single /24 to advertise into the IGRP domain which uses a /24 mask. This command is entered on the ASBR between the OSPF and the IGRP (or

RIP) domain. Again, the summarization must occur on the router that is redistributing into OSPF (i.e., creating the OSPF external routes).

Stub and NSSA Areas

When configuring a stub or NSSA area, all routers in the area must agree on the stub or NSSA.

Area Type	Gets External routes?	Gets Inter-Area Routes?	Gets a default route?	Can generate Ext Routes?
area 1 stub	no	yes	yes	no
area 1 stub no-summary	no	no	yes	no
area 1 nssa	no	yes	no	yes
area 1 nssa no-summary	no	no	yes	yes

Use a stub area (`area 1 stub`) to block external (type 1 and type 2) routes from being sent to the stub area. Use a stub area with no summary (`area 1 stub no-summary`) to block all OSPF routes except those from within that area (this commands blocks inter-area routes, external type-1 routes and external type-2 routes).

Use an NSSA area when you want to block external (type 1 or type 2) routes from being sent to the area (NSSA areas do not get OSPF external routes) but you want the area to be able to originate external routes, such as from redistribution. NSSA external routes can be summarized by the router that connects between the NSSA area and the backbone.

Virtual Links

When setting up virtual links, the area defined is the area through which the virtual link will traverse. You do need to have every OSPF ABR touch area 0, either directly or through a virtual link. When configuring the virtual link, you must use the ***router id*** of the other end of the virtual link.

If area 0 is using authentication, you must add either the `authentication-key` or `message-digest-key` to the `area n virtual-link` command. Additionally you must add `area 0 authentication [message-digest]` to all routers in area 0 including the router at the "far" end of the virtual link (even though it doesn't really "touch" area 0 – it only connects via the virtual link).

For the following network:

```
R1 --- area 0 --- R2 --- area 1 --- R3 --- area 2 --- R4 --- area 3
```

R3 needs a virtual link to R2 and R4 needs a virtual link to R3.

Prefix Lists

The way prefix lists work are you can specify a network and mask or a network and a range of masks. Specifying a network and mask is fairly simple:

```
ip prefix-list mylist seq 10 permit 172.16.25.0/24
```

This will allow (match) the exact network 172.16.25.0/24 to pass the list. However prefix lists can also specify a network with a range of masks. For example:

```
ip prefix-list mylist seq 10 permit 172.16.0.0/16 ge 24 le 26
```

This will take the entire class B network 172.16.0.0 (172.16.0.0/16) and pass only subnets with a /24, /25 or /26 mask (ge 24 le 26). So the exact network 172.16.0.0/16 would actually fail the list because it does not have a mask of /24, /25 or /26.

By default if you only specify “ge” then any subnet with a mask greater than or equal to the ge value will pass. That is, ge all the way up to /32. For example:

```
ip prefix-list mylist seq 10 permit 10.10.10.0/24 ge 28
```

This list specifies any subnet within the 10.10.10.0/24 range that has a mask of /28 or greater (255.255.255.240 → 255.255.255.255). Again, the exact subnet 10.10.10.0/24 would fail because it does not have a mask of /28 or greater.

By default if you only specify “le” then any subnet with a mask less than or equal to the le value but greater than or equal to the mask specified will pass. That is, le all the way down to the mask listed. For example:

```
ip prefix-list mylist seq 10 permit 10.64.0.0/16 le 23
```

This list specifies any subnet within the 10.64.0.0/16 range that has a mask between /16 and /23, inclusive (255.255.0.0 → 255.255.254.0). In this case the exact subnet 10.64.0.0/16 would pass because it has a mask in the range /16 → /23.

The “permit any any” in a prefix list is:

```
ip prefix-list mylist seq 200 permit 0.0.0.0/0 le 32
```

Redistribution

When redistributing into EIGRP or IGRP by using the `redistribute` command you need to explicitly configure the EIGRP or IGRP metric (otherwise no routes get redistributed). You can do this by:

```
router eigrp 1
 default-metric 1000 50 255 128 1514
```

or

```
router eigrp 1
 redistribute bgp 65222 metric 1000 50 255 128 1514
```

Note: in EIGRP the metric values are BW (in Kbits/sec), delay, reliability, loading and MTU

When redistributing OSPF into BGP it appears BGP will only accept OSPF internal (inter- and intra-area) routes – **not external type 1 or type 2** routes by default. To change this, use:

```
router bgp 65000
  redistribute ospf 1 match internal external 1 external 2
```

When redistributing ISIS into another protocol, you may need to explicitly include `level-1-2` for all routes to get redistributed. It appears level-2 ISIS routes are what are redistributed by default.

When redistributing into RIP, make sure you add the `metric` keyword, such as `metric 3`, otherwise you may get the metric set to 16 (unreachable).

When you redistribute, make sure that you don't "violate" the requirement of summarization. For example, you may be summarizing OSPF routes. You may also be required to run EIGRP on those interfaces and redistribute these into OSPF. If you don't use a route-map to control which routes get placed into OSPF, you'll see the OSPF summary and external OSPF routes for each of the EIGRP interfaces. For example, you may be asked to summarize:
172.16.8.0/24 (e0/0)
172.16.9.0/24 (e0/1)
172.16.10.0/24 (s0/0)

So you create an area range 172.16.8.0/22 statement. However you also need to run EIGRP on the 172.16.0.0 network (thus EIGRP gets run on all the above interfaces). For full connectivity, you have to redistribute EIGRP into OSPF. When you do that, all of the three routes listed above go back into OSPF as external routes. Thus other OSPF routers will have the OSPF area summary, but also the specific routes as OSPF externals. To prevent this, filter (route-map) on the redistribution of EIGRP into OSPF.

Route Maps

My recommendation is to become extremely proficient with route maps. It is an incredibly powerful tool. You will typically need them during redistribution since you are usually limiting what routes get redistributed. However they can also perform a myriad of other functions: setting almost any BGP attribute, setting route tags, setting various routing parameters (metric, metric-type, etc.) filtering routes inbound or outbound from BGP neighbors, performing policy routing, controlling various IPX functions, etc. I typically used these for most of my filtering functions. Even though they may be an extra command out two (compared to a distribute-list) I was so used to using them it was more comfortable to use route maps. Practice route maps!!

Remember, when working with route maps the behavior is as follows:

- For policy routing, if none of the route-map statements match the packet gets routed normally
- For routing updates if none of the route-map statements match the routing update gets dropped

Thus if you are using a route-map to modify some routing updates (set communities, set tags, etc.) in order to still propagate all other routing updates, you need:

```
route-map mymap permit 200
  (nothing here – this is left blank)
```

Router "Network" Statements

When you specify a network via the `network` statement in eigrp, rip, etc. that triggers the software to run that protocol on those connected routes (usually interfaces) within that range and to incorporate that network into the protocol's database.

However connected routes also include static routes that use a next-hop interface (if you look via `sho ip route` a static route with a next hop of an interface shows as "connected").

Split Horizon

Many routing protocols use split horizon. Often split horizon is turned off on a physical Frame-Relay interface. Often on a remote router you will want to turn this on.

Often split horizon is turned on on a Frame-Relay point-to-point or multipoint subinterface. Often on the hub router you will want to turn this off.

It is a good practice when using a protocol that runs split-horizon (IP RIP, IP IGRP, IP EIGRP, IPX RIP, IPX EIGRP), to manually set the split-horizon to the way you need it, regardless of the default.

You can not turn IPX RIP split horizon off! If you need to route IPX over NMBA (Frame Relay or ATM) it needs to be EIGRP or NLSP with split horizon disabled on the hub router.

Tips & Tricks

The lab should provide colored pens and pencils. However these are about the only thing you actually can bring into the lab with you. It might be a good idea to bring good erasers, pens and sharpened, colored pencils.

Remember that `control-shift-6 control-shift-6` will not break to the terminal server but rather send a break to the actual router the terminal server is connected to. This is very handy. For example, you can set up an extended ping of 1000 pings (that are failing) so that you can troubleshoot what is happening on other routers (shows, debugs, etc.) Once you have solved or identified the issue (or given up!) go back to the router doing the pinging and type `control-shift-6 control-shift-6` to break from the extended ping.

If you have a serial cross-over cable and you don't know which end is DCE or DTE, connect each end to routers and do:

```
show controllers serial 0
```

Access Lists

Often you'll get asked to do things with access-lists in the "fewest number of lines." Watch this closely. Don't just use the fewest number of "permit" statements necessary. Often its an interesting combination of denies and permits. For example:

Use an access-list to allow from 172.16.32.0 to 172.16.247.255 (inclusive). You could do:
access-list 1 permit 172.16.32.0 0.0.31.255
access-list 1 permit 172.16.64.0 0.0.63.255
access-list 1 permit 172.16.128.0 0.0.63.255
access-list 1 permit 172.16.192.0 0.0.31.255
access-list 1 permit 172.16.224.0 0.0.15.255
access-list 1 permit 172.16.240.0 0.0.7.255

However you could accomplish this with fewer lines:
access-list 1 deny 172.16.0.0 0.0.31.255
access-list 1 deny 172.16.248.0 0.0.7.255
access-list 1 permit 172.16.0.0 0.0.255.255

For access-lists:

- BGP uses TCP port 179
- RIPv1 uses UDP port 520
- OSPF uses protocol 89 and dest. address 224.0.0.5
- EIGRP uses protocol 88
- IGRP uses ?
- DLSw uses TCP 2065 and 2067 and – if prioritization is used – TCP 1981, 1982, 1982.
- ESP (IPSec) uses protocol 50
- AH (IPSec) uses protocol 51
- GRE uses protocol 47
- ISAKMP uses UDP port 500

For netbios host name access lists, `permit *` is the "permit any."
For mac-address lists `permit 0000.0000.0000 ffff.ffff.ffff` is the "permit any."

At the end of an access-list place a "deny any any log" to send rejected packets to the log. This will help determine what packets may be getting blocked that are causing other things (routing protocols, tunnels, ipsec, etc.) not to work.

Terminal Editing

`control-A` brings you to the beginning of the line
`control-E` brings you to the end of the line
`control-R` repaints a line (handy if a console message pops up)
`control-U` is the same as "up arrow" (in case that isn't working)

Tunnels

You basically indicate a source and a destination IP address. Then assign whatever characteristics you want to the tunnel (bridging, IPX network, etc.)

If a router learns about it's tunnel destination address over the tunnel it will try to send the GRE (or whatever tunnel mode you are using) packets over the tunnel itself... that won't work! Use a distribute list to prevent each side from advertising it's tunnel source address to the other side over the tunnel. For example, perhaps two routers have a tunnel between their loopback addresses. Let's say they are using RIP to be able to route between the loopbacks. If OSPF is enabled on the tunnel and the loopbacks, OSPF will deliver updates (through the tunnel) about the loopback networks. Since OSPF has a better admin distance than RIP, it will supercede the RIP learned routes.

Yet now the router is attempting to maintain the tunnel (route packets to the destination) through the tunnel – but it can't maintain the tunnel if the "next hop" is inside the tunnel!

In this case you will usually get a console message that the tunnel interface is down due to a routing loop.