

ACRC Study Notes 12/12/98

1. Describe the key requirements of a scalable internetwork

Reliable
Responsive
Efficient
Adaptable
Accessible but secure

2. Select a Cisco IOS feature as a solution for a given internetwork requirement

Reliable:	routing protocols that increase reachability and convergence, eg. OSPF Consider redundant links
Responsive	queuing
Efficient	optimize bandwidth, drop broadcasts etc Establish connections when necessary, eg DDR Reduce amount of routing update overhead traffic
Adaptable	cope with routable and non-routable traffic eg bridging Use integrated routing eg IGRP
Secure	authentication to and from each remote site Secure access to network devices themselves, eg. Line vty access.

3. Describe causes of network congestion

Bursts of user traffic
Multicast / broadcast overhead traffic, eg routing updates, SAP adverts, keepalives
Too much traffic on low bandwidth links
Network design issues

4. List solutions for controlling network congestion

Filter unwanted traffic
Reduce overhead - adjust timers for routing announcements, provide static entries
Prioritize traffic

5. Configure IP standard access lists

Used to permit or deny a complete protocol suite based on the source
network/subnet/host address
Are in the range of 1-99 (IP).
Are processed top-down, i.e. first matching rule preempts further
processing

Only one access list per port per direction per protocol
Uses a wildcard mask to define which bits of the network address are relevant (0 = check this bit, 1 = ignore [inverted subnet mask])
Place standard lists close to the destination, place extended list close to the source

```
access-list <number> (permit|deny) <src> [<src wildcard>]
```

```
int eth0  
  ip access-group <number> (in|out)
```

6. Limit virtual terminal access

```
Rt1[config]# line vty 0 4  
Rt1[config-line]# access-class <access-list-number> (in | out)
```

Filters inbound telnet sessions to the router or outbound sessions from the router.
Access list can be any standard or extended IP access list.

7. Configure IP extended access lists

```
access-list <number> (permit|deny) prot src src-wildcard src-port dst dst-wildcard dst-port [est] [log]
```

```
int eth0  
  ip access-group <number> (in|out)
```

8. Verify access list operation

```
show ip interfaces (to see applied access lists)  
show access-lists <number>  
clear access-list counters
```

```
show line (to see lists on telnet lines)
```

9. Configure an alternative to using access lists

Use the null interface and route unwanted traffic to that interface to save CPU cycles processing access lists.

Define the null interface:

```
Rt1[config]# interface null0  
  
Rt1[config]# ip route <address> <mask> null 0
```

Interface name is always null0.

10. Configure an ip helper address to manage broadcasts

Forwards certain broadcasts. When forwarding it usually changes the destination from being a broadcast to that of unicast helper address. Though the helper address can actually be a broadcast in which case it is broadcasted on the appropriate interface. Multiple helper addresses can be specified.

```
Rt1[config-if]# ip helper-address <address>
```

ip forward-protocol is for forwarding packets that aren't forwarded by default. Default protocols are BOOTP, tftp, DNS, time, NetBIOS Name service, NetBIOS datagram, TACAS.

```
Rt1[config]# ip forward-protocol { udp [port] | nd | sdns }
```

nd Forward Network Disk (ND) datagrams. This protocol is used by older diskless Sun workstations.
sdns Secure Data Network Service.

11. Describe IPX/SPX traffic management issues

Overhead traffic: watchdog, SPX keepalive, SAP, RIP, NLSP, GNS. Use traffic, SAP, NLSP, GNS filters.

Static SAP addresses in SAP table:

```
ipx sap <service-type> <server name> <address> <socket> <hop-count>
```

<service-type> is file server, print server, NMS, etc.

<socket> is like TCP ports, or 0 for all. Routing, ping use sockets.

e.g:

```
ipx sap 107 MAILSERV 160.0000.0c01.2b72 8104 1
ipx sap 4 FILESERV 165.0000.0c01.3d1b 451 1
ipx sap 143 JOES_SERVER A1.0000.0c01.1234 8170 2
no route to A1, JOES_SERVER won't be announced until route is learned
```

107 is RCONSOLE, 4 is file server. Socket 451 is NCP. Socket 8xxx is Novell Server processes.

The router responds to GNS queries, and this command specifies that it responds with each server name evenly since GNS only gets one server.

```
ipx gns-round-robin
```

GNS – Get Nearest Server

See CCNA notes for IPX configuration information.

12. Filter IPX traffic using IPX access lists

Standard lists are 800-899, extended lists 900-999, SAP lists are 1000-1099
The Cisco router does not forward SAP broadcasts, but constructs its own SAP table and broadcasts that (every 60 seconds [standard])
A -1 for service/port specifies 'any' service/port.

Standard access-list: (use number 800-899)

Basically filter on network or node address only.

```
access-list <number> (deny|permit) src-net[.srcnode[srcnodemask]] [dst-net[.dstnode][dstnodemask]]
```

```
int eth0
  ipx access-group <list-number> {in | out}
```

Extended access list: (use number 900-999)

Basically has protocol and socket numbers. Also log option.

```
access-list <number> (deny|permit) <protocol> [src-net[.srcnode][srcnetmask.srcnodemask]] src-socket [dst-net[.dstnode][dstnetmask.dstnodemask]] dst-socket [log]
```

```
int eth0
  ipx access-group <list-number> {in | out}
```

<socket> is like TCP ports, or 0 for all. Routing, ping use sockets.
<protocol> is RIP, SAP, SPX, NCP, NetBIOS, or -1 for any.

Examples

The following example denies access to all RIP packets from the RIP process socket on source network 1 that are destined for the RIP process socket on network 2. It permits all other traffic. This example uses protocol and socket names rather than hexadecimal numbers.

```
access-list 900 deny -1 1 rip 2 rip
access-list 900 permit -1
```

SAP: (use number 1000-1099)

To create list -

```
access-list <number> (permit|deny) network[.node] [network-mask node-mask] [service type[server-name]]
```

Apply filter(s) to interface -

```

int eth0
ipx input-sap-filter <number> (filter before constructing SAP table)
ipx output-sap-filter <number> (filter when sending SAP table)
ipx router-sap-filter <number> (filter adverts from a certain router)

```

SAP Input Filter Example

SAP input filters allow a router to determine whether to accept information about a service.

Router C1, illustrated in [Figure 21](#), will not accept and, consequently not advertise, any information about Novell server F. However, Router C1 will accept information about all other servers on the network 3c. Router C2 receives information about servers D and B.

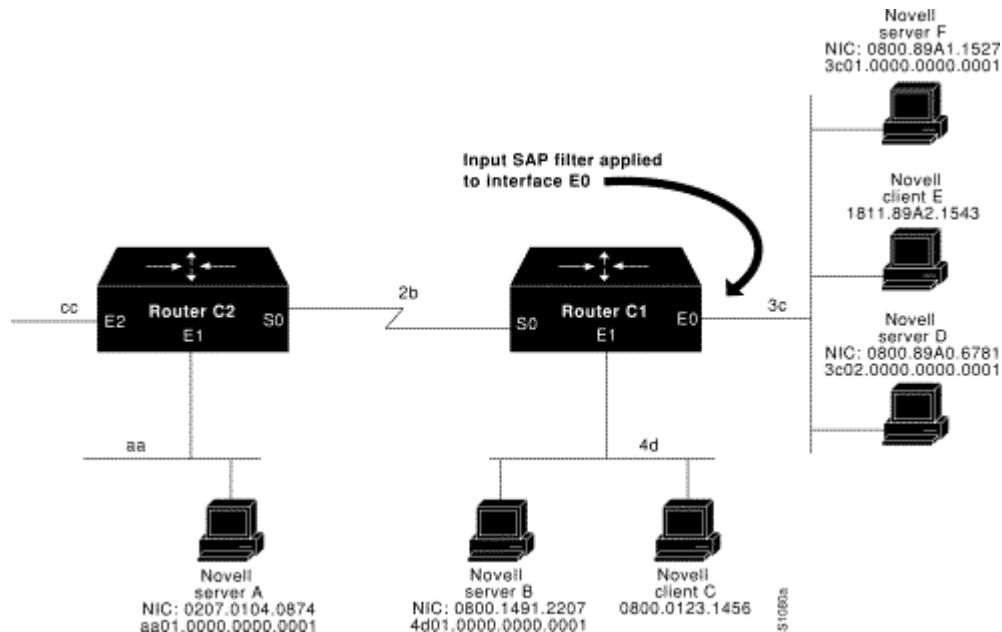


Figure 21: SAP Input Filter

The following example configures Router C1. The first line denies server F, and the second line accepts all other servers.

```

access-list 1000 deny 3c01.0000.0000.0001
access-list 1000 permit -1
interface ethernet 0
ipx network 3c
ipx input-sap-filter 1000
interface ethernet 1
ipx network 4d
interface serial 0
ipx network 2b

```

Note NetWare Versions 3.11 and later use an internal network and node number as their address for access list commands (the first configuration command in this example).

Example

The following access list blocks all access to a file server (service Type 4) on the directly attached network by resources on other Novell networks, but allows access to all other available services on the interface:

```

access-list 1001 deny -1 4

```

```
access-list 1001 permit -1
```

GNS filters:

Uses SAP access list type. Filters responses the router gives to client GNS queries.

```
Rt1[config-if]# ipx output-gns-filter <access-list>
```

13. Manage IPX/SPX traffic over WAN connection

Two options, IPX directly over the WAN link using IPXWAN, or tunneling IPX over an IP tunnel.

```
Rt1[config]# interface serial 0  
Rt1[config-if]# ipx ipxwan
```

Use IPXWAN with PPP or HDLC encapsulation. Nodes on the link introduce themselves and negotiate delay, bandwidth etc. To use the command on an interface you must not have specified a ipx network number using the `ipx network` command on that interface

```
.  
Rt1[config-if]# ipx ipxwan [<local-node> {<network-num>|unnumbered}]
```

Unnumbered is the default and 0 is the default for <local-node>, which sets it to the internal network number which is used by NLSP as well and can be set with the following command:

```
Rt1[config]# ipx internal-network-number <network-number>
```

Otherwise the local-node number is a network number not used anywhere else in the network and the router with the highest number is the master with respect to IPXWAN link negotiations.

The <network-num> parameter specifies the network number for the shared link if that router is the master (i.e. it will tell the slave router this address is the network number). Unnumbered sets it to 0.

There's also local-server-name and retry-interval and retry-limit parameters as well.

What to do when the serial link fails:

```
Rt1[config-if]# ipx ipxwan error {reset | resume | shutdown}
```

Use static routes (both routers need this set):

```
Rt1[config-if]# ipx ipxwan static
```

IPX tunnel over IP:

```
interface tunnel 0
```

```

ipx network <own-network-number>
tunnel source serial 0
tunnel destination <remote-ip-number>
tunnel mode gre ip
no ip address

```

gre ip is generic routing encapsulation over IP.

Set interval for advertising SAP table over a WAN link:

```

Rt1[config-if]# ipx sap-interval <minutes>

```

14. Verify IPX/SPX filter operation

```

show ipx interfaces
show access-lists <number>
show ipx access-lists <number>

```

15. Describe the need for queuing in a large network

Delay sensitive applications may require priority, e.g. telnet.
Most effective on T1 links and below that experience temporary congestion.
To provide acceptable service levels and control WAN costs
Only needed on congested links.
Requires buffering in the router.

Monitoring commands:

```

show queuing
show queuing custom

```

Weighted Fair Queuing	Priority Queuing	Custom Queuing
no queue lists	4 queues	16 queues
low volume given priority	high queues serviced first	round robin service
conversation dispatching	packet dispatching	threshold dispatching
interactive traffic gets priority	critical traffic gets through	allocation of available bandwidth
file transfers get balanced access	designed for low bandwidth links	best suited for high bandwidth link
enabled by default		

16. Describe weighted fair queuing operation

Used if no strict control is needed by just 'fair' use of a link.

Weighted fair queuing can manage duplex data streams, such as those between pairs of applications, and simplex data streams such as voice or video. From the perspective of weighted fair queueing, there are two categories of sessions: high-bandwidth sessions and low-bandwidth sessions. Low-bandwidth traffic has effective priority over high-bandwidth traffic, and high-bandwidth traffic shares the transmission service proportionally according to assigned weights.

When weighted fair queuing is enabled for an interface, new messages for high-bandwidth traffic streams are discarded after the configured or default congestive-messages threshold has been met. However, low-bandwidth conversations, which include control-message conversations, continue to enqueue data. As a result, the fair queue may occasionally contain more messages than its configured threshold number specifies.

Enabled by default on links 2Mb or below.

Not used on LAPB, X25, SDLC links and not supported on tunnels.

```
Rt1[config-if]# fair-queue [<congestive-discard-threshold> [<dynamic  
queues> <reservable queues>]]
```

Default: congestive-discard-threshold: 64 messages; dynamic-queues: 256; reservable-queues: 0.

Congestive discard threshold is the number of messages allowed in each queue in the range 1-4096. When the number of messages in the queue for a high-bandwidth conversation reaches the specified threshold, new high-bandwidth messages are discarded.

Dynamic queues are used for best effort conversations.

Reservable queues are used for RSVP.

17. Configure priority queuing

Uses four different queues – high, medium, normal, low – and you specify which traffic goes into which queue.

1. Create priority list
2. Associate a default queue.
3. Specify queue sizes (optional)
4. Assign a list to an interface.

Create priority list and prioritize traffic based on protocol or incoming interface:

```
priority-list <number> protocol <protocol-name> { high | medium |  
normal | low } [<queue-keyword> <keyword-value>]
```

<number> can be from 1 to 16.

Protocol is ip, appletalk, ipx, etc.

Queue keywords are:

fragment	specify fragmented packets, not including first fragment
gt <byte-count> ,	
lt <byte-count>	specify packets of a certain range in size
list <list-number>	specify an access list to specify the traffic
tcp <port> , udp <port>	specify traffic to certain ports

```
priority-list <number> interface <interface-type> <interface-number> {  
high | medium | normal | low }
```

Specify the default queue any non specified packets go into:

```
priority-list <list-number> default { high|medium|normal|low }
```

The maximum number packets that can be waiting in each queue:

```
priority-list <number> queue-limit <hi-limit> <medium-limit> <normal-  
limit> <low-limit>
```

A value of 0 means unlimited. Default is 20, 40, 60, 80. If a queue overflows, excess packets are discarded and quench messages can be sent, if appropriate, for the protocol.

Assign the list to an interface:

```
Rt1[config-if]# priority-group <list-number>
```

Example

```
priority-list 2 protocol ip high tcp 23  
priority-list 2 protocol ip high list 1  
priority-list 2 interface ethernet 0 medium  
priority-list 2 protocol ip normal  
priority-list 2 default low  
priority-list 2 queue-limit 15 20 20 30  
  
access-list 1 permit 131.108.0.0 0.0.255.255  
  
interface serial 0  
    priority-group 2
```

18. Configure custom queuing

1. Set custom queuing filtering for protocols or interfaces
2. Assign a default queue.
3. Change queue capacity (optional)
4. Configure the transfer rate per queue.
5. Assign the custom queue-list to an interface.

Create queue list:

```
queue-list <list-number> protocol <protocol-name> <queue-number>
[<queue-keyword> <keyword-value>]
```

<list-number> can be from 1 to 16. <queue-number> can be from 1 to 16.

Protocol is ip, appletalk, ipx, etc.

Queue keywords are:

gt <byte-count> ,	
lt <byte-count>	specify packets of a certain range in size
list <list-number>	specify an access list to specify the traffic
tcp <port> , udp <port>	specify traffic to certain ports

```
priority-list <list-number> interface <interface-type> <interface-
number> <queue-number>
```

Specify default queue:

```
queue-list <list-number> default <queue-number>
```

Change the queue capacity (maximum number of packets in queue):

```
queue-list <list-number> queue <queue-number> limit <limit-number>
```

Change the size of each queue (size in bytes):

```
queue-list <list-number> queue <queue-number> byte-count <byte-count-
number>
```

Assign queue list to an interface:

```
custom-queue-list <list-number>
```

Example:

```
queue-list 1 protocol ip 1 tcp 20
queue-list 1 protocol ip 2
queue-list 1 protocol ipx 3
queue-list 1 protocol appletalk 4
```

```
queue-list 1 default 5
queue-list 1 queue 1 byte-count 4500
```

```
interface serial 0
 custom-queue-list 1
```

19. List the key information routers need to route data

Routing table and the packets destination.

20. Compare distance vector and link-state protocol operation

Distance vector protocols periodically swap routing table information with neighbouring routers. Link State protocols swap link state information to all routers in the area.

21. Given an IP address, use VLSMs to extend the use of the IP address

This is just subnetting and supernetting/CIDR. Use a variable length network prefix.

22. Given a network plan that includes IP addressing, explain if route summarization is or is not possible

Route summarization is just using VLSMs to specify a group of subnets and is used by IGRP and OSPF. CIDR is used by BGP4. Also known as route aggregation.

23. Define private addressing and determine when it can be used

Use of 10.0.0.0 and 172.16.0.0 and 192.168.0.0 (private) networks cannot directly talk to the Internet.

24. Define network address translation and determine when it can be used

Two of the key problems facing the Internet are depletion of IP address space and scaling in routing. Network Address Translation (NAT) is a feature that allows an organization's IP network to appear from the outside to use different IP address space than what it is actually using. Thus, NAT allows an organization with nonglobally routable addresses to connect to the Internet by translating those addresses into globally routable address space. NAT also allows a more graceful renumbering strategy for organizations that are changing service providers or voluntarily renumbering into CIDR blocks. NAT is also described in RFC 1631.

As a solution to the connectivity problem, NAT is practical only when relatively few hosts in a stub domain communicate outside of the domain at the same time. When this is the case, only a small subset of the IP addresses in the domain must be translated into globally unique IP addresses when outside communication is necessary, and these addresses can be reused when no longer in use.

A significant advantage of NAT is that it can be configured without requiring changes to hosts or routers other than those few routers on which NAT will be configured. As discussed previously, NAT may not be practical if large numbers of hosts in the stub domain communicate outside of the domain. Furthermore, some applications use embedded IP addresses in such a way that it is impractical for a NAT device to translate. These applications may not work transparently or at all through a NAT device. NAT also hides the identity of hosts, which may be an advantage or a disadvantage.

If the software cannot allocate an address because it has run out of addresses, it drops the packet and sends an ICMP Host Unreachable packet.

A router configured with NAT must not advertise the local networks to the outside. However, routing information that NAT receives from the outside can be advertised in the stub domain as usual.

There is normal NAT with static or dynamic (from a pool) address translation.

There is also ‘overloading’ of an inside global address which is like IP masquerading under Linux.

There is translation of overlapping address which is used when the inside network is using an address space used on the Internet. What the router does is check all incoming DNS query responses and checks to see if an outside DNS server is responding with the internally used addresses and if so it modifies the packet and basically does a reverse NAT type thing. See documentation.

There is also TCP load distribution where outside to inside traffic to say a web server can be load balanced by having the router dynamically translate to a server in a pool. Five web servers internally but one outside IP address.

Configure Static Translation

Task	Command
Establish static translation between an inside local address and an inside global address.	ip nat inside source static <i>local-ip global-ip</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

Configure Dynamic Translation

To configure dynamic inside source address translation, perform the following tasks beginning in global configuration mode:

Task	Command
Define a pool of global addresses to be allocated as needed.	ip nat pool <i>name start-ip end-ip {netmask netmask prefix-length prefix-length}</i>
Define a standard access list permitting those addresses that are to be translated.	access-list <i>access-list-number</i> permit <i>source [source-wildcard]</i>
Establish dynamic source translation, specifying the access list defined in the prior step.	ip nat inside source list <i>access-list-number</i> pool <i>name</i>
Specify the inside interface.	interface <i>type number</i>
Mark the interface as connected to the inside.	ip nat inside
Specify the outside interface.	interface <i>type number</i>
Mark the interface as connected to the outside.	ip nat outside

25. Explain why OSPF is better than RIP in a large internetwork

OSPF is:

- Scalable
- Faster convergence
- VLSM support
- Handles big networks (more than 15 hops),
- Less network traffic and is multicast
- Better path determination.

26. Explain how OSPF discovers, chooses, and maintains routes

Discover neighbors:

RouterA is enabled on the LAN and is in the 'down' state.

RouterA sends a hello packet via multicast

Other routers add the RouterA data to their adjacency database. This is the 'Init' state.

Other routers reply with unicast hello packets and RouterA adds their data to its own adjacency database. Now that all routers have complete adjacency databases, this is the 'two-way' state.

Election of DR/BDR:

Router with highest priority is the DR

Router with the next highest priority is the BDR.

Ties are resolved by making DR the router with the highest Router ID.

Default priority is 1 and can be set. A priority of 0 means the router cannot be the DR/BDR.

High Router IDs can be set with loopback interfaces with high IP addresses.

Swap routes:

After an election the routers are in 'Exstart' state ready to swap LSAs.

DR and BDR establish adjacencies with each router in the network (e.g. LAN) and a master/slave relationship is created, the router with the highest ID is the master.

Master and slave routers exchange one or more database description packets (DBDs OR DDPs). This is the 'Exchange' state.

A DBD includes LSAs which have information about a link or network. including link type, the advertising router, cost of the link, and a sequence number.

Slave acknowledges the DBD with a LSAck.

Slave then compares DBD entries with its own database and if DBD has more up-to-date entry the slave sends a LSR – Link State Request.

Master responds to a LSR with an LSU which the slave LSACKs.

Sending LSRs is the 'Loading' state.

All routers add new entries to their link-state database.

After all LSRs are satisfied and a routing table has been created the adjacent routers are in 'Full' state and can then route traffic and should have identical databases.

Route selection:

Route selection is based on link costs and the Dijkstra's SPF algorithm.

spf holdtime is a hold timer (default 10 seconds) that the router will delay SPF recalculations in case a link is going up and down lots (flapping).

Routers send hello packets every 10 seconds (default) to its neighbor to check the link state.

When a link goes down:

The router that noticed sends a multicast LSU to all DR/BDRs on 224.0.0.6.

DR acks the LSU and floods the LSU onto the network using 224.0.0.5. Each router sends an LSACK in receipt of the LSU.

Routers update their link state database and recalculate SPF routing.

If a router receives an old LSU it will send back its newer LSU.

LSA entries in the link-state database have an aging timer (default 30 minutes) after which the router that originated the entry sends an LSU on the network to verify the link.

27. Configure OSPF for proper operation

Enable OSPF and specify networks to include:

```
Rt1[config]# router ospf <process-id>
```

```
Rt1[config-router]# network <address> <wildcard-mask> area <area-id>
```

<wildcard-mask> is a mask like in access lists. 1s don't care, 0s means bits count.

Create a loopback interface if you want to specify a router ID:

Rt1[config]# interface loopback <number>

Set link priority:

Rt1[config-if]# ip ospf priority <number> (0 – 255)

Set a link cost:

Rt1[config-if]# ip ospf cost <cost> (1 – 65535)

Link cost is usually based on the speed of the link – the bandwidth. The bandwidth can be set with the bandwidth command since the default maybe wrong. e.g. serial defaults to 1.544Mbps. Cost is based on 10^8 / bandwidth.

28. Verify OSPF operation (single-area)

```
show ip protocol
show ip route
show ip ospf interface
show ip ospf
show ip ospf neighbor detail
show ip ospf database
```

```
debug ip ospf
```

29. Describe the issues with interconnection multiple areas and how OSPF addresses each

If a very large network not divided into areas, then get:
Frequent SPF calculations when a link goes up/down.
Large routing table
Large link-state table.

To fix these problems divide the network into areas and use route summarization, so LSUs are generally only intra-area.

30. Explain the differences between the possible types of areas, routers, and LSAs

Routers: internal, backbone, ABR, ASBR

Areas: standard, backbone, stub, totally stubby

LSAs: type

- 1 router link entry
- 2 network link (from a DR about the multi-access it's a DR of, e.g. a LAN
- 3 summary link (from a ABR about it's area and distributes on the backbone where the other ABRs flood their non-totally stubby areas.

- 4 summary link (about ASBR reachability)
- 5 AS external link (from a ASBR about external routes
 - E1 – includes cost of internal links – use if more than one ASBR
 - E2 – doesn't have cost of internal links – no point if there is only one ASBR.

31. Configure a multi area OSPF network

ABR is nothing special other than having two or more network area commands.

Route summarization:

Route summarization on an ABR:

```
Rt1[config-router]# area <area-id> range <address> <mask>
```

Route summarization on an ASBR:

```
Rt1[config-router]# summary-address <address> <mask>
```

<address> is the route summary address, and <mask> is the network prefix for the summary route.

Stub Areas:

Command for all routers:

```
Rt1[config-router]# area <area-id> stub [ no-summary ]
```

except [no-summary] is for ABRs on totally stub areas – i.e. no summary of external routes.

For ABRs, define the cost of the default route that is injected into stubby area:

```
Rt1[config-router]# area <area-id> default-cost <cost>
```

Virtual links:

All areas are supposed to be connected to the backbone area. If an area can't be directly connected to the backbone area than it can be tunneled via a virtual link through a transit area. Both routers at each end of the virtual link need:

```
Rt1[config-router]# area <area-id> virtual-link <router-ID>
```

where <area-id> is the transit area and router-id is the id of the virtual link neighbor (show ip ospf will display a routers ID – usually it's IP address).

Transit areas cannot be stub areas.

32. Verify OSPF operation (multi-area)

```
show ip ospf border-routers
show ip ospf virtual-links
show ip ospf <process-id>
show ip ospf [<process-id> <area-id>] database ?
```

33. Describe Enhanced IGRP features and operation

Features:

Rapid convergence using Diffusing Update Algorithm (DUAL). Stores backup routes, or if there aren't any sends out a query. Routers not affected by the change don't have to recompute routing tables.

Reduced bandwidth – no periodic updates like RIP and updates only sent to neighbors and not flooded on the network.

Multiple Network layer support – IP, IPX, Appletalk.

Novell

- Incremental updates of RIP and SAP tables default on WAN links.
- Maximum is 224 hops (not 15)
- Optimal path selection based on bandwidth and delay not hops and ticks.
- Redistribution of NLSP.

Appletalk:

- Better metrics
- No periodic updates
- Use in clientless environment because clients only know RTMP.

Operation:

RouterA multicasts a hello packet to discover neighbors.

Neighbor routers put their route information from their topology database (except split horizon type) and set an Init bit on the packet. The information includes metrics and the router that advertised it.

RouterA send an Ack to the neighbors

RouterA then exchanges update packets with each of its neighbors who send an Ack in reply.

After all updates are received it chooses a primary route and up to 6 backup routes.

EIGRP supports internal, external and summary routes.

Criteria is usually the smallest bandwidth in a path and the cumulative delay in a path.

Not usually used is reliability, load, and MTU.

Advertised distance is the distance from the next-hop router to the destination.

Feasible distance is the distance between the local router and the next-hop router.

The primary path is the one with the lowest cost total of advertised and feasible distance.

The next hop router is known as the successor and there can be more than one.

The backup path or feasible successor is a next hop router whose advertised distance is less than the successors feasible distance.

Unreachable routes (metric infinity) are removed from the topology database.

Updates:

Hello packets are sent every 5 seconds (or 60 seconds in a NBMA network).

When a router learns of a change it multicasts the update to its neighbors.

If a feasible successor is in the topology table it is promoted and replacement backup routes are looked for based on the new successors feasible distance.

If no feasible successor exists the router flags the route as 'active' and then sends a query to its neighbors.

Neighbor will send a reply if it has no route or if it has a feasible route which doesn't go through the querying router, else it too sends a query to its neighbors....

After all replies to the query have come back it either has a new route or removes the active route.

If a neighbor that was queried doesn't reply then it is considered down and a update is sent out as above...

34. Configure Enhanced IGRP

IP:

Enable EIGRP with the AS number of the local AS:

```
Rt1[config]# router eigrp <as-number>
```

Add connected networks that will be advertised:

```
Rt1[config-router]# network <network-number>
```

Set bandwidth on interfaces for a more realistic metric:

```
Rt1[config-if]# bandwidth <kilobits>
```

IP route summarization:

Disable automatic route summarization by EIGRP if you have discontinuous networks:

```
Rt1[config-router]# no auto-summary
```

If you want to manually summarize an address space out an interface:

```
Rt1[config-if]# ip summary-address eigrp <as-num> <address> <mask>
```

IPX:

```
Rt1[config]# ipx routing  
Rt1[config-ipx-router]# ipx router { eigrp <as-number> | rip }  
Rt1[config-router]# network <network-number>
```

Under the RIP section disable the networks that will be distributed via EIGRP:

```
Rt1[config-router]# no network <network-number>
```

To set incremental SAP updates (i.e. no SAP update traffic on an interface unless there is a change in the SAP table:

```
Rt1[config-if]# ipx sap-incremental eigrp <as-number> [rsup-only]
```

rsup specifies that just sap-incremental is used on this interface and no EIGRP.

35. Verify Enhanced IGRP operation

```
show ip eigrp neighbours  
show ip eigrp topology  
show ip route eigrp  
show ip protocols  
show ip eigrp traffic  
  
show ipx route  
show ipx eigrp neighbours  
show ipx eigrp topology
```

36. Select and configure the different ways to control route update traffic

Use passive interface:

Go into the specific routing protocol section and then:

```
Rt1[config-router]# passive-interface <interface>
```

Disables all outgoing updates on the specified interface.

Use default routes:

```
Rt1[config]# ip default-network <network>
```

```
Rt1[config]# ipx advertised-default-route-only <network>
```

Where <network> specifies the destination network of where all the default packets go.

For IGRP and RIP use ip classless to enable these protocols to allow classless forwarding, i.e any subnets without a route will be sent to the default route. RIP and IGRP are usually classful and if no routes is specified for a subnet in a specific class C,B,A network the packet is dropped.

```
Rt1[config]# ip classless
```

If there was no default network then ip classless would send the packet to the best supernet route.

Use static routes:

```
Rt1[config]# ip route <prefix> <mask> {<address>|<interface>}  
[<distance>] [permanent]
```

where <distance> is the administrative distance of the route (default is 1 for a static route). permanent specifies if route stays in routing table even if interface goes down.

```
Rt1[config]# ipx route {<network>|default} {<network.node> |  
<interface>} [floating-static]
```

A floating static route can be overridden by a dynamically learned route.

Under a certain routing protocol section you can do the following to advertise static routes:

```
Rt1[config-router]# redistribute static
```

Use routing updates access lists:

To filter incoming route updates (either all updates or just ones in on an interface)

```
Rt1[config-router]# distribute-list <access-list> in [<interface>]
```

To filter outgoing route updates (either by interface, routing protocol or AS):

```
Rt1[config-router]# distribute-list <access-list> out [ <interface> |  
<routing-process> | <as-number> ]
```

37. Configure route redistribution in a network that does not have redundant paths between dissimilar routing processes

Access the routing protocol into which you want routes redistributed:

e.g.

```
Rt1[config]# router ospf
```

Specify the protocol to be redistributed (protocol can be static or connected or ospf etc.):

```
Rt1[config-router]# redistribute <protocol> [<process-id>] [metric  
<value>] [metric-type <value>] [subnets]
```

subnets is required for OSPF for redistributing subnets.

Define seed metric that the receiving router uses to calculate the value of the route before redistributing:

When redistributing IGRP or EIGRP:

```
Rt1[config-router]# default-metric <bandwidth> <delay> <reliability>  
<loading> <mtu>
```

When redistributing RIP, OSPF, BGP:

```
Rt1[config-router]# default-metric <number>
```

To avoid loops set the default metric to one larger than the largest native metric.

38. Configure route redistribution in a network that has redundant paths between dissimilar routing processes

39. Resolve path selection problems that result in a redistributed network

Modify administrative distance for the particular protocol. Go into the appropriate protocol:

```
Rt1[config-router]# distance <weight> [<address> <mask> [access-list]]
```

where <weight> is the new administrative distance. <access-list> is the selection of networks to be redistributed with the new administrative distance.

40. Verify route redistribution

Check routing tables:

```
show ip route  
show ipx route  
etc.
```

Trace routes, debug routing activity:

trace
debug

41. Describe when to use BGP to connect to an ISP

Use BGP when ISP and local AS use different policies.

Use when local AS needs information about other ASes in the network.

If there are two connections to the Net that are always up (i.e. not a backup line) and being used then you need BGP.

Else static routes will do the trick.

42. Describe methods to connect to an IP using static and default routes, and BGP.

Static routes:

Set a default route:

```
Rt1[config]# ip route 0.0.0.0 0.0.0.0 <interface>  
Rt1[config]# ip route 0.0.0.0 0.0.0.0 <address>
```

Make sure there is a route to the ISP.

When IGP is OSPF, redistribute the default route into the OSPF AS using

```
Rt1[config]# default-information originate always
```

BGP:

Enable BGP using a local BGP AS number assigned by InterNIC:

```
Rt1[config]# router bgp <AS-number>
```

All networks you want to advertise:

```
Rt1[config-router]# network <network-number>
```

Specify BGP neighbors and peers (peers use the local BGP AS-number):

```
Rt1[config-router]# neighbor <address> remote-as <AS-number>
```

Clear the BGP information when you make a BGP configuration change:

```
router# clear ip bgp *
```

Verify BGP operation:

```
show ip bgp
show ip bgp paths
show ip bgp summary
show ip bgp neighbors
```

43. Compare the differences between WAN connection types: dedicated, asynchronous dial-in, dial-on-demand, and packet-switched services

44. Determine when to use PPP, HDLC, LAPB, and IETF encapsulation types

PPP is for point to point link or dial-up links.

HDLC is for point to point links

LAPB is for packet switched public networks

IETF is a parameter for frame relay encapsulation and is for non Cisco router communication.

```
Rt1[config-if]# encapsulation hdlc
Rt1[config-if]# encapsulation ppp
Rt1[config-if]# encapsulation lapb [dte | dce] [multi | <protocol>]
```

Multi specifies that multiple protocols can be carried in LAPB frame, else <protocol> specifies the protocol that can be carried.

Compression:

Uncompressed normal

Header compression Header, e.g. ip tcp-headercompression

Payload compression Data, e.g. frame-relay payload-compress

Link compression Whole frame, e.g. compress [predictor | stac]

WAN switching types:

Frame relay	X.25	SMDS	ATM
frame based	packet based	packet switching	cell switching
56k – 45Mbps	56k – T1	DS1 – DS3	upto 155Mbps
Connection	Connection	Connectionless	Connectionless
PVC + SVC	PVC + SVC		PVC + SVC
Error checking. No retransmission	Extensive error checking and retransmission	No error checking	No error checking

45. List at least four common issues to be considered when evaluating a WAN service

Availability
Bandwidth
Cost
Ease of management
Application traffic
Routing protocol characteristics

46. Describe the components that make up ISDN connectivity

Functions are devices or hardware functions:

- TA - Terminal adapter
- TE1|2 - Terminal end point 1 (integrated TA) or 2 (needs TA)
- NT1 - Network termination ("signal converter")
- NT2 - found in PBXs etc, combines layer 2 and 3 functions
- NT1/NT2 combined units
- LT - Local termination (access point at phone company)
- ET - Exchange termination (communicates with other ISDN components)

Reference points are interfaces (between functions)

Phone --- TA --- NT1 --- ISDN Switch
R S/T U

PC --- NT1 --- ISDN Switch
S/T U

Bearer channel: Used for data transfer (voice or data)
Data channel: Used for control/signaling information using LAPD

47. Configure ISDN BRI

```
isdn switch-type <type> (telco dependant)
ip route <network> <mask> <next-hop> (setup static route)
dialer-list <group> protocol <prot-name> [permit|deny] (access list)

int bri0
encapsulation [ppp|hdlc]
ip address <address> <mask>

dialer-group <group-number>
dialer idle-timeout 300
dialer-map <protocol> <next-hop-address> [name <name>] [speed <speed>]
<dial-string> [broadcast]

isdn spid1 <number> (to set up Service Profile ID, provider specific)
isdn spid2 <number> (same for 2nd bearer channel)
```

Rate adaptation is the setting of the speed parameter in the dialer-map command.

Sub addresses are used if there is more than one device connected to the NT1 eg fax/phone. Use :X appended to the 'phone number', where X is the device number.

Bandwidth on demand - use load-threshold to specify when to use extra channel via multilink PPP. Can be based on traffic (in | out | either).

Call screening use isdn caller <number>
Number which interface responds to: isdn answer1 <called-party-number>
Both can use sub-addresses.

The following example enables Multilink PPP on BRI 0:

```
interface BRI0
description Enables PPP Multilink on BRI 0
ip address 7.1.1.1 255.255.255.0
encapsulation ppp
dialer map ip 7.1.1.2 name starbuck 14195291357
dialer map ip 7.1.1.3 name roaster speed 56 14098759854
ppp authentication chap
ppp multilink
dialer load-threshold either 10
dialer-group 1
```

Monitoring ISDN	
Show controller bri00	bri unit and D channel info
Show interface bri0	interface info
Show dialer	dialer group info

48. Configure legacy dial-on-demand router (DDR)

1. Define interesting traffic
2. Enable DDR on the interface
3. Assign traffic definition to an interface
4. Define destinations
5. Configure call parameters

1. Define interesting traffic

```
Rt1[config]# dialer-list <group> protocol <prot-name> [permit|deny]
or
router[config]# dialer-list <group> list <access-list-number>
```

2. Enable DDR on the interface:

```
Rt1[config-if]# dialer in-band (not needed for BRI)
```

3. Assign traffic definition to an interface:

```
Rt1[config-if]# dialer-group <group-number>
```

4. Define destinations:

```
dialer-map <protocol> <next-hop-address> [name <name>] [speed <speed>]  
        <dial-string> [broadcast]
```

If no dialer map is specified:

```
dialer-string <dial-string>
```

5. Configure call parameters:

```
dialer load-threshold <load> {inbound | outbound | either }
```

Where load is an integer between 1 and 255. bandwidth command has to specify the bandwidth for the interface so it can calculate the load correctly (well that's what the documentation says).

```
dialer idle-timeout <seconds>
```

```
dialer fast-idle <seconds>
```

Timeout when there's contention for the line.

```
dialer wait-for-carrier <seconds>
```

```
dialer hold-queue <packets>
```

Rotary groups:

Create a dialer interface used in the rotary configuration. It has to use PPP and you set all the parameters for all the physical interfaces on this logical interface, i.e. the physical interfaces will all have identical setups.

```
Rt1[config]# interface dialer <number>  
Rt1[config-if]# encapsulation ppp
```

Place physical interfaces (serial, async, BRI) in the rotary group:

```
Rt1[config-if]# dialer rotary-group <number>
```

To set if one interface should be used first give it a high priority (255 is highest):

```
Rt1[config-if]# dialer priority <0-255>
```

Example:

```
ip route 171.68.12.0 255.255.255.0 131.108.126.2  
ip route 171.68.14.0 255.255.255.0 131.108.126.3  
dialer-list 2 protocol ip permit
```

```

username 2500A password cisco
username 2500B password cisco

interface dialer 3
ip address 131.108.126.1 255.255.255.0
encapsulation ppp
ppp authentication chap
dialer map ip 131.108.126.2 name 2500A 4779624
dialer map ip 131.108.126.3 name 2500B 4811942
dialer-group 2
dialer load-threshold 160
dialer fast-idle 15
dilaer idle-timeout 120

interface bri0
  dialer rotary-group 3
...
interface bri3
  dialer rotary-group 3

```

49. Configure dialer profiles

```

interface dialer 1
ip address
encapsulation ppp
dialer pool 4
dialer group 1
dialer string 54355434

```

where the bri is setup as:

```

interface bri0
no ip address
encapsulation ppp
dialer pool-member 4 priority 255
ppp authentication chap

```

50. Verify DDR operation

```

show dialer interface bri0
show dialer map
ping
telnet
show isdn active
show isdn status
show ip route

```

51. Configure dial backup

On the router interface you want to backup:

```
Rt1[config-if]# backup interface <backup-interface-name>
```

Note when using a BRI interface as a backup the BRI can't be used for other things.

Specify load limits for backup use:

```
Rt1[config-if]# backup load {<enable-threshold> | never } { <disable-load> | never }
```

Enabled threshold is the load before the secondary is enabled. Disable load threshold is the combined received load or combined transmitted load on both lines at which point the secondary will be disabled if traffic is below the threshold.

Set response timers for up/down:

```
Rt1[config-if]# backup delay {<enable-delay>| never } {<disable-delay> | never}
```

Enable delay is the delay after the line has gone down that the backup is enabled. The disable delay is the time after the primary line has come up that the secondary is disabled.

Suggested values are 20 seconds to wait to see if it is actually down, and wait 5 seconds to see that the link is staying up.

52. Verify dial backup operation

Pull the plug? Probably will show up on a show interface.

53. Configure Multilink PPP

```
ppp multilink
dialer load-threshold <load> {inbound | outbound | either }
```

54. Verify Multilink PPP

```
show dialer
debug ppp multilink
show isdn
show dialer interface
```

55. Configure snapshot routing

Snapshot routing is during specified quiet times there is no routing activity and there is no normal traffic then the link will be down. Though if there is no traffic for a specified length of time then a client router will dialup a server router and exchange routing information for another specified amount of time (active-time). When the link is down

Client:

Specify a BRI interface
Configure the client router

```
Rt1[config-if]# snapshot client <active-time> <quiet-time> [suppress-  
exchange-updates] [dialer]
```

<active-time> is from 5 to 100 minutes, typically 5 minutes.

<quiet-time> is from 8 to 100000, has to be at least <active-time> + 3 minutes.

suppress-exchange-updates disables the exchange of routing tables each time link comes up – ie only do routing exchanges on the specified active/quiet time setup.

dialer specifies whether the clients dials up when the link is down due to no normal traffic.

Define a dialer map

```
Rt1[config-if]# dialer map snapshot <sequence-number> name <name>  
<dial-string>
```

<sequence-number> is a unique number from 1 to 254 which prioritizes the dialer map if there is more than one map.

<name> is the server name

<string> is the number for the server

Server:

Specify the ISDN interface
Configure the server router

```
Rt1[config-if]# snapshot server <active-timer> [dialer]
```

<active-timer> must be the same as the client.

dialer specifies whether the clients dials up when the link is down due to no normal traffic.

Verify Snaphost routing:

```
show snapshot
```

```
clear snapshot quiet-time bri0
```

```
show snapshot
```

(will now notice state is now active)

```
debug snapshot
```

```
debug ip rip
```

56. Configure IPX spoofing

Let the router spoof clients and servers by spoofing keepalives and watchdogs when the link is down.

Turn off route caching (fast-switching) so that the router has to look inside packets to determine it's contents:

```
Rt1[config]# no ipx route-cache
```

```
Rt1[config]# ipx spx-spoof
```

```
Rt1[config]# ipx watchdog-spoof
```

Optionally – the default is 60 seconds:

```
Rt1[config]# ipx spx-idle-time
```

This command sets the elapsed time in seconds after which spoofing of keepalive packets occurs, following the end of data transfer; that is, after the acknowledgment and sequence numbers of the data being transferred have stopped increasing. By default, SPX keepalive packets are sent from servers to clients every 15 to 20 seconds.

If you turn on SPX spoofing and you do not set an idle time, the default of 60 seconds is assumed. This means that the dialer idle time begins when SPX spoofing begins. For example, if the dialer idle time is 3 minutes, the elapse time before SPX spoofing begins is 4 minutes: 3 minutes of dialer idle time plus 1 minute of SPX spoofing idle time.

57. Define routable and nonroutable protocols and give an example of each

Routable protocols have network addressing information at layer 3 (network), non-routable protocols don't.

Examples are IP and Netbeui, LAT, MOP.

58. Define various bridging types and describe when to use each type

Transparent Ethernet

Encapsulation: for bridging non-routable protocols over FDDI or serial links

IRB: Router does both bridging and routing

 Migrate a bridged network to a routed network

 Connect a remote site that doesn't have routing capabilities

 Conserve addresses

 Increase performance by keeping local bridged traffic local.

SRB: Token ring.

SRT a mix of SRB and Transparent, but no translation. The interface will bridge packets without RIF (transparent) and packets with RIF (SRB). Token ring.

SR/TLB tries to do some translation. Perhaps more common for between Ethernet and Token Ring.

FDDI rings can only bridge transit Ethernet traffic i.e. traffic sourced from an Ethernet network with a destination of another Ethernet network and not a host on the FDDI ring.

59. Configure transparent bridging

Select spanning tree protocol and assign interfaces to bridge group.

```
Rt1[config]# bridge <bridge-group> protocol ( ieee | dec )
Rt1[config-if]# bridge-group <bridge-group>
```

Can assign a priority to the bridge – for spanning tree protocol.

Can specify a cost for an outgoing interface.

```
Rt1[config]# bridge <bridge-group> priority <number>
Rt1[config-if]# bridge-group <bridge-group> path-cost <cost>
```

Basic Bridging Example

[Figure 34](#) is an example of a basic bridging configuration. The system has two Ethernets, one Token Ring, one FDDI port, and one serial line. The IP is being routed, and everything else is being bridged. The Digital-compatible bridging algorithm with default parameters is being used.

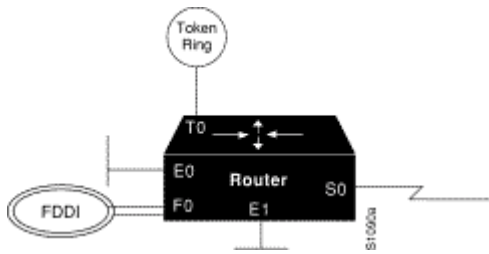


Figure 34: Example of Basic Bridging

The configuration file for the router depicted in [Figure 34](#) would be as follows:

```
interface tokenring 0
  ip address 131.108.1.1 255.255.255.0
  bridge-group 1
!
interface fddi 0
  ip address 131.108.2.1 255.255.255.0
  bridge-group 1
!
interface ethernet 0
  ip address 192.31.7.26 255.255.255.240
  bridge-group 1
!
```

```

interface serial 0
  ip address 192.31.7.34 255.255.255.240
  bridge-group 1
!
interface ethernet 1
  ip address 192.31.7.65 255.255.255.240
  bridge-group 1
!
bridge 1 protocol dec

```

```

show bridge
show span

```

The following are possible variations of the **show bridge** command:

```

show bridge ethernet 0
show bridge 0000.0c00.0000 0000.00FF.FFFF
show bridge 0000.0c00.0e1a
show bridge
show bridge verbose

```

In the sample output, the first command would display all entries for hosts reachable via Ethernet interface 0, the second command would display all entries with the vendor code of 0000.0c00.0000, and the third command would display the entry for address 0000.0c00.0e1a. In the fourth command, all entries in the forwarding database would be displayed. The fifth command provides additional detail. In all five lines, the bridge-group number has been omitted.

Sample Displays

The following is sample output of the **show bridge** command. The second display is output from the **show bridge** command with the **verbose** argument.

Router# **show bridge**

Total of 300 station blocks, 280 free

Codes: P - permanent, S - self

Bridge Group 32:Bridge Group 32:

Address	Action	Interface	Age	RX count	TX count
0180.c200.0000	receive	-	S	0	0
ffff.ffff.ffff	receive	-	S	0	0
0900.2b01.0001	receive	-	S	0	0
0300.0c00.0001	receive	-	S	0	0
0000.0c05.1000	forward	Ethernet0/1	4	1	0
0000.0c04.4b5b	receive	-	S	0	0
0000.0c04.4b5e	receive	-	S	0	0
0000.0c04.4b5d	receive	-	S	0	0
0000.0c04.4b5c	receive	-	S	0	0
0000.0c05.4a62	forward	Ethernet0/1	4	1	0
aa00.0400.2108	forward	Ethernet0/1	0	42	0
0000.0c12.b888	forward	Ethernet0/2	4	1	0
0000.0c12.b886	forward	Ethernet0/1	4	1	0
aa00.0400.4d09	forward	Ethernet0/1	4	1	0
0000.0c06.fb9a	forward	Ethernet0/1	4	1	0
0000.0c04.b039	forward	Ethernet0/1	4	1	0

RouterA# **show span**

Bridge Group 1 is executing the DEC compatible Spanning Tree protocol

Bridge Identifier has priority 128, address 0000.304c.f686


```

Configured hello time 1, max age 15, forward delay 30
We are the root of the spanning tree
Topology change flag not set, detected flag not set
Times: hold 1, topology change 30, notification 30
        hello 1, max age 15, forward delay 30, aging 300
Timers: hello 1, topology change 0, notification 0

```

```

Port 7 (ATM0.1 LANE Ethernet) of bridge group 1 is down
Port path cost 0, Port priority 128
Designated root has priority 128, address 0000.304c.f686
Designated bridge has priority 128, address 0000.304c.f686
Designated port is 7, path cost 0
Timers: message age 0, forward delay 0, hold 0

```

60. Configure Integrated Routing and Bridging (IRB)

Router has routed interfaces and bridged interfaces – a protocol cannot be bridged and routed on the same interface. What can be done is a virtual interface is created which represents a bridge group and can be included as a routed interface. The interface is just like any other standard router interface (e.g. can have filters etc.) and is called the Bridge-group Virtual Interface – BVI . For packets received on the bridge interface(s) if the destination MAC address is that of the routers, then the router routes the packet, else it consults it's bridging table. Packets can be routed from router interface out through the bridging interface. BVI gets a MAC address from one of the bridge group's interfaces.

First configure the interfaces, routing and bridge groups as normal, then configure the BVI interface.

```

bridge 20 ieee
int e0
    bridge-group 20
int e1
    bridge-group 20

```

```

bridge irb                                enable IRB
interface bvi 20                          create BVI
    ip address 10.0.0.1                   give it an address
bridge route (ip | appletalk | etc)      specify routing

```

IRB bridges all protocols by default. Can disable bridging of specific protocols with

```
no bridge 20 bridge ip
```

May also need to redefine the bvi's MTU and encapsulation types.

Cannot do IRB: Cisco 7000 and AGS+ routers, ISDN, X.25, SRB

Cannot do IRB at the same time as concurrent bridging.

IRB only does IP, IPX, Appletalk.

Do not configure protocol attributes on the bridge interfaces when both routing and bridging a given protocol.

Do not configure bridging attributes on the BVI.

```
show interface bvi 1
show interfaces eth 2 irb
```

Basic Integrated Routing and Bridging Example

[Figure 35](#) is an example of integrated routing and bridging that uses bridge group 1 to bridge and route IP. The router has three bridged Ethernet interfaces and one routed Ethernet interface.

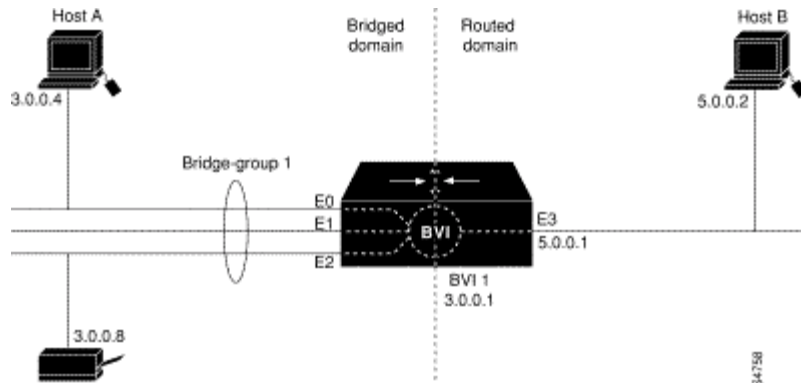


Figure 35: Basic IP Routing using Integrated Routing and Bridging

The relevant portions of the configuration for the router are listed below.

```
interface Ethernet 0
  bridge-group 1
!
interface Ethernet 1
  bridge-group 1
!
interface Ethernet 2
  bridge-group 1
!
interface Ethernet 3
  ip address 5.0.0.1 255.0.0.0
!
interface BVI 1
  ip address 3.0.0.1 255.0.0.0
!
bridge irb
bridge 1 protocol ieee
bridge 1 route ip
```

61. Describe the basic functions of source-route bridging (SRB)

For routing packets between rings. Source learns the route to the destination. It firsts test to see if destination is local with a 'test' frame. If destination is local then bridges aren't used. Else, if the test fails the source sends out an 'explorer' frame which all bridges forward on all their interfaces adding routing information to the frame as they forward it so that the path to the destination will be recorded. The destination then sends back all received frames. The source then chooses from all the returned frames which path to use. It's main criteria in path choice is: order frames arrived, number of hops to destination, maximum MTU along the path, or a combination.

In the frames there is a bit called the RII – routing information indicator. It is the most significant bit in the source MAC address and is set by the source.

62. Configure SRB

Specify local bridge connections and enable spanning explorers

```
[config-if]# source-bridge <local-ring> <bridge-number> <target-ring>
[config-if]# source-bridge spanning
```

SRB-Only Example

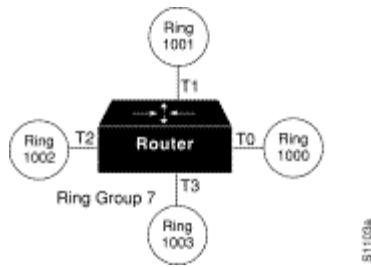
The following example shows that all protocols are bridged, including IP. Because IP is being bridged, the system has only one IP address.

```
no ip routing
!
interface tokenring 0
 ip address 131.108.129.2 255.255.255.0
 source-bridge 129 1 130
 source-bridge spanning
!
interface tokenring 1
 ip address 131.108.129.2 255.255.255.0
 source-bridge 130 1 129
 source-bridge spanning
!
interface ethernet 0
 ip address 131.108.129.2 255.255.255.0
```

Multi-port configuration (i.e. more than two interfaces). Create a virtual ring inside the router and then routing between multiple rings is done through the virtual ring. On each interface the source bridge target ring parameter changes to the virtual ring.

```
Rt1[config]# source-bridge ring-group <ring-group-number>
```

SRB Multiport Example



The following is a sample configuration file:

```

source-bridge ring-group 7
!
interface tokenring 0
  source-bridge 1000 1 7
  source-bridge spanning
!
interface tokenring 1
  source-bridge 1001 1 7
  source-bridge spanning
!
interface tokenring 2
  source-bridge 1002 1 7
  source-bridge spanning
!
interface tokenring 3
  source-bridge 1003 1 7
  source-bridge spanning

```

Router can limit the explorer packets on the network by changing the explorer packet (broadcast) to a specific frame (unicast) and adding the routing information of the destination node if the router has the required routing information in its tables. The destination will then respond back to the source.

```
Rt1[config-if]# source-bridge proxy-explorer
```

Enables the collection and use of routing information to let you bridge routable protocols over a source-bridged network:

```
Rt1[config-if]# multiring { <protocol> | all | other }
```

The type of network that would want multiring:



Multiring Example

These commands enable IP and Novell IPX bridging on a Token Ring interface. RIFs will be generated for IP frames, but not for the Novell IPX frames.

```
! commands that follow apply to interface token 0
interface tokenring 0
! generate RIFs for IP frames
multiring ip
! enable the Token Ring interface for IP
ip address 131.108.183.37 255.255.255.0
! enable the Token Ring interface for Novell IPX
novell network 33
```

Enable automatic spanning tree

```
[config-if]# source-bridge spanning <bridge-group>
```

To set path cost for the interface for the spanning tree:

```
[config-if]# source-bridge spanning <bridge-group> path-cost <cost>
```

Enable IBM bridging protocol (optional):

```
[config]# bridge <bridge-group> protocol ibm
```

63. Configure source-route transparent bridging (SRT)

Same as normal transparent bridging but generally uses ieee bridging protocol. If there is source-routed traffic then source-bridge commands would be used – i.e. bridge-group commands on Ethernet interfaces and source-bridge command on token ring interfaces.

This seems optional:

For SRT bridging networks, you must choose the OUI code that will be used in the encapsulation of Ethernet Type II frames across Token Ring backbone networks. To choose the OUI, perform the following task in interface configuration mode:

```
ethernet-transit-oui [90-compatible | standard | cisco]
```

64. Configure source-route translational bridging (SR/TLB)

To enable Source-route Translational Bridging a virtual ring (like multiport SRB setup) is created. The transparent bridging group is then treated like a pseudo ring and added to the ring group. So other rings think the transparent bridge group is just another ring. The transparent bridge group thinks the virtual ring is just another bridge group. The router does translation between the two frame formats – RIF information added or removed.

source-bridge transparent <ring-group> <pseudo-ring> <bridge-number> <transapnet-bridge-group> [oui]

I.e. this command adds a transparent pseudo group to the ring group.
oui is organization unit identifier.

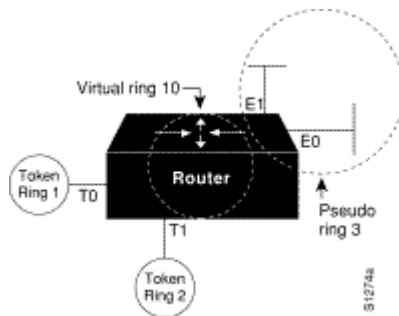
SR/TLB handles MTU, RIF, bit ordering, and MAC addresses embedded in the data area e.g an ARP frame.

Multiple paths between the SRB and transparent bridge groups are not allowed due to possible loops.

SR/TLB for a Simple Network Example

In the simple example illustrated in [Figure 61](#), a four-port router with two Ethernets and two Token Rings is used to connect transparent bridging on the Ethernets to SRB on the Token Rings.

Figure 61: Example of a Simple SR/TLB Configuration



Assume that the following configuration for SRB and transparent bridging existed before you wanted to enable SR/TLB:

```
interface tokenring 0
  source-bridge 1 1 2
!
interface tokenring 1
  source-bridge 2 1 1
!
interface ethernet 0
  bridge-group 1
!
interface ethernet 1
  bridge-group 1
```

```
!  
bridge 1 protocol dec
```

To enable SR/TLB, one aspect of this configuration must change immediately--a third ring must be configured. Before SR/TLB, the two Token Ring interfaces were communicating with two-port local source-route bridging; after SR/TLB, these two interfaces must be reconfigured to communicate through a virtual ring, as follows:

```
source-bridge ring-group 10  
!  
interface tokenring 0  
    source-bridge 1 1 10  
!  
interface tokenring 1  
    source-bridge 2 1 10  
!  
interface ethernet 0  
    bridge-group 1  
!  
interface ethernet 1  
    bridge-group 1  
!  
bridge 1 protocol dec
```

Now you are ready to determine two things:

- A ring number for the pseudo-ring that is unique throughout the source-route bridged network. For the preceding example configuration, use the number 3.
- A bridge number for the path to the pseudo-ring. For the preceding example configuration, use the number 1.

Once you have determined the ring number and the bridge number, you can add the **source-bridge transparent** command to the file, including these two values as parameters for the command. The following partial configuration includes this **source-bridge transparent** entry:

```
!  
source-bridge ring-group 10  
source-bridge transparent 10 3 1 1  
!  
interface tokenring 0  
    source-bridge 1 1 10  
!  
interface tokenring 1  
    source-bridge 2 1 10  
!  
interface ethernet 0  
    bridge-group 1  
!  
interface ethernet 1  
    bridge-group 1  
!  
bridge 1 protocol dec
```

65. Verify SRB operation

```
show source-bridge
show rif
```

```
Router# show source-bridge
Local Interfaces:
      srn bn   trn r p s n   max hops   receive   transmit   drops
TR0      39:1002      23:62923      5      1      10 *      *
Ring Group 10:
  This peer: TCP 150.136.92.92
  Maximum output TCP queue length, per peer: 100
Peers:
  TCP 150.136.92.92      -      2      0      0      0      0      0
  TCP 150.136.93.93      open  2*     18     18     3      0      0
Rings:
  bn: 1 rn: 5      local ma: 4000.3080.844b TokenRing0      fwd: 18
  bn: 1 rn: 2      remote ma: 4000.3080.8473 TCP 150.136.93.93      fwd: 36
Explorers: ----- input ----- ----- output -----
      spanning all-rings      total      spanning all-rings      total
TR0      0      3      3      3      5      8
Router#
```

```
Router# show rif
Codes: * interface, - static, + remote
Hardware Addr How Idle (min) Routing Information Field
5C02.0001.4322 rg5 - 0630.0053.00B0
5A00.0000.2333 TR0 3 08B0.0101.2201.0FF0
5B01.0000.4444 - - -
0000.1403.4800 TR1 0 -
0000.2805.4C00 TR0 * -
0000.2807.4C00 TR1 * -
0000.28A8.4800 TR0 0 -
0077.2201.0001 rg5 10 0830.0052.2201.0FF0
```

In the display, entries marked with an asterisk (*) are the router's interface addresses. Entries marked with a dash (-) are static entries. Entries with a number denote cached entries. If the RIF timeout is set to something other than the default of 15 minutes, the timeout is displayed at the top of the display.

[Table 23](#) describes significant fields shown in the display.

Table 23: Show RIF Field Descriptions

Field	Description
Hardware Addr	Lists the MAC-level addresses.
How	Describes how the RIF has been learned. Possible values include a ring group (rg), or interface (TR).
Idle (min)	Indicates how long, in minutes, since the last response was received directly from this node.
Routing Information F	Lists the RIF.

Notes:

E1/T1 Configuration – See ACRC slides.

BPDU

Bridge Protocol Data Unit. Spanning-Tree Protocol hello packet that is sent out at configurable intervals to exchange information among bridges in the network.

PPP

```
username <name> password <password> (to set CHAP password)
```

```
int ser0
  encapsulation ppp
  ppp authentication (pap | chap)
  ppp multilink
```

Example

Router LEFT:

```
hostname LEFT
username RIGHT password cisco
```

```
interface bri0
ip address 131.109.126.1 255.255.255.0
encapsulation ppp
ppp authentication chap
dialer map ip 121.109.126.2 name RIGHT 4779625
```

Router RIGHT:

```
hostname RIGHT
username LEFT password cisco
```

```
interface bri0
ip address 131.109.126.2 255.255.255.0
encapsulation ppp
ppp authentication chap
dialer map ip 121.109.126.1 name LEFT 4811943
```

Add RIP to configuration

```
router rip
  network xx.xx.xx.xx
  network yy.yy.yy.yy
```

The network command specifies the interfaces that RIP info is sent and received. Also only those interfaces will be advertised.

```
debug ip rip          (for debugging)
```

Add IGRP to conf

```
router igrp <autonomous system number>
  network xx.xx.xx.xx
  network yy.yy.yy.yy
```

The network command specifies the interfaces that IGRP info is sent and received. Also only those interfaces will be advertised.

42) List problems each routing type encounters when dealing with topology changes and describe techniques to reduce the number of these problems

Distance Vector limitations:

Time to convergence, e.g. distance vector (hop-count) updates occur from router to router (not broadcast)

PROBLEM: Routing loops (slow detection of topology change, incorrect routing table/updates)

SOLUTION 1: Split horizon - If you learn a route on one interface do not send back that info on that same interface - so as not to cause loops.

SOLUTION 2: Route poisoning - If net unreachable set route metric to infinity.

PROBLEM: Network topology keeps changing giving inconsistent network.

SOLUTION: Hold-down timers - Ignore updates for a set period so as not to screw up convergence if an interface goes up and down many times in a short period.

PROBLEM: Counting to infinity (incorrect routing table/updates)

SOLUTION: Define a maximum metric ("max. hop count +1 = infinity"). If metrics get too high due to routing loops.

Link state routing problems:

Processing power, e.g. link-state (OSPF) needs lots of processing power to rebuild the routing database (tree)

Network bandwidth, e.g. initial (multicast) link-state info floods the network

PROBLEM: Unsynchronised updates can arrive at different time based on bandwidth (slow and fast links)

SOLUTION: Use time stamps, update numbering and counters

PROBLEM: Synchronisation of large networks

SOLUTION: Use link state area hierarchy for topology

SOLUTION: Exchange route summaries at area borders

PROBLEM: Bandwidth and resources usage

SOLUTION: Dampen the periodic update (longer intervals)

SOLUTION: Use targeted multicast (not flood)

PROBLEM: partitioned regions - slow parts separated from fast parts

SOLUTION: manage network using area hierarchy

Frame relay

10) List commands to configure Frame Relay LMIs, maps, and subinterfaces

```
int ser0
  encapsulation frame-relay [ietf | cisco]    (default = cisco)
```

Optional:

```
  frame-relay lmi-type [ansi | cisco | q933a] (default = cisco)
  keepalive 10
```

Static mapping of dlci:

```
  frame-relay map <protocol> <address> <dlci> [broadcast] [ietf|cisco]
```

(map a DLCI to a network for a given protocol, allows routing updates only if "broadcast" is defined: frame map ip 192.168.0.1 70. Used if other router doesn't use Inverse ARP, or the protocol doesn't support it.)

Dynamic mapping of dlci doesn't need anything - just uses Inverse ARP.

```
int s0.<sub-number> (point-to-point | multi-point)
  frame-relay interface-dlci <dlci> (broadcast)
```

The following example illustrates how to configure two routers for static mode.

Configuration for Router 1

```
interface serial 0
  ip address 131.108.64.2 255.255.255.0
  encapsulation frame-relay
  keepalive 10
  frame-relay map ip 131.108.64.1 43
```

Configuration for Router 2

```
interface serial 0
  ip address 131.108.64.1 255.255.255.0
  encapsulation frame-relay
  keepalive 10
  frame-relay map ip 131.108.64.2 43
```

Basic Subinterface Examples

```
interface serial 0
  encapsulation frame-relay
interface serial 0.1 point-to-point
  ip address 10.0.1.1 255.255.255.0
  frame-relay interface-dlci 42
!
interface serial 0.2 multipoint
  ip address 10.0.2.1 255.255.255.0
  frame-relay map 10.0.2.2 18
```

Broadcast queuing:

Creates a queue for the specified interface to hold broadcast traffic that's replicated for output on multiple DLCIs.

```
frame-relay broadcast-queue <size> <byte-rate> <packet-rate>
```

Default is size 64 packets, 256000 bytes per second and 36 packets per second.

Frame Relay Switching:

Create Frame Relay networks using Cisco routers. Can create a switching network or tunnel frame relay.

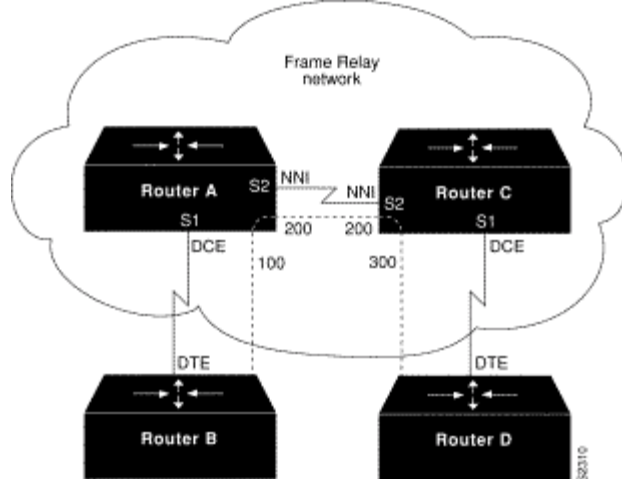
```
Rt1[config]# frame-relay switching
Rt1[config-if]# frame-relay route <in-dlci> <out-interface> <out-dlci>
Rt1[config-if]# frame-relay intf-type [ dtc | dce | nni ]
```

nni – Network to Network Interface for inter-switch communication.

Pure Frame Relay DCE Example

Using the PVC switching feature, it is possible to build an entire Frame Relay network using our routers. In the following example, Router A and Router C act as Frame Relay switches implementing a two-node network. The standard Network-to-Network Interface (NNI) signaling protocol is used between Router A and Router C (see [Figure 20](#)).

Figure 20: Frame Relay DCE Configuration



Configuration for Router A

```

frame-relay switching
!
interface ethernet 0
 no ip address
 shutdown
!
interface serial 0
 ip address 131.108.178.48 255.255.255.0
 shutdown
!
interface serial 1
 no ip address
 encapsulation frame-relay
 frame-relay intf-type dce
 frame-relay lmi-type ansi
 frame-relay route 100 interface serial 2 200
!
interface serial 2
 no ip address
 encapsulation frame-relay
 frame-relay intf-type nni
 frame-relay lmi-type q933a
 frame-relay route 200 interface serial 1 100
 clockrate 2048000

```

Configuration for Router C

```

frame-relay switching
!
interface ethernet 0
 no ip address
 shutdown :Interfaces not in use may be shut down; shut down is not required.
!
interface serial 0
 ip address 131.108.187.84 255.255.255.0
 shutdown
!
interface serial 1
 no ip address
 encapsulation frame-relay
 frame-relay intf-type dce
 frame-relay route 300 interface serial 2 200
!

```

```
interface serial 2
no ip address
encapsulation frame-relay
frame-relay intf-type nni
frame-relay lmi-type q933a
frame-relay route 200 interface serial 1 300
```

11) List commands to monitor Frame Relay operations in the router

show int ser0	show DLCI and LMI information
debug frame-relay lmi	
show frame-relay lmi	LMI statistics
show frame-relay pvc	PVC statistics
show frame-relay traffic	router's global FR statistics
show frame-relay map	
show frame-relay router	display static routes