CS 638 Lab 4: Intra-Domain Routing

Joe Chabarek, Mike Blodgett and Paul Barford
University of Wisconsin –Madison
jpchaba,mblodget,pb@cs.wisc.edu

Layer 3 (the network layer) is the so-called "narrow waste" of the Internet since there is only one protocol that is defined at this layer – the Internet Protocol (IP). The foundation for the network layer is spelled out in the now classic paper written in 1974 by Vinton Cerf and Robert Kahn entitled *A Protocol for Packet Network Interconnection.*

Activities that take place at the network layer occur on end hosts (in particular sending nodes, which place source and destination IP addresses in packets), and on devices that reside on end-to-end paths (*i.e.,* routers). When a packet that is destined for an end host that is geographically far away leaves its local area network, it will encounter devices that are active at layer 3. These devices are fundamentally concerned with forwarding packets between networks based on IP addresses.

Routers are designed to operate at two different levels. The *data plane* in a router is concerned with the task of packet forwarding. Data plane activities typically take place on line cards in routers and must be done at very high speed sufficient to keep up with the maximum signaling rates available on the device (*i.e.,* the bandwidth of links). Data planes of today's high speed routers also offer a host of specialized capabilities for improving performance, quality and manageability.

The *control plane* of the router is concerned with configuration and monitoring of the system and most importantly with the routing. Routing is the process by which forwarding tables for a device are established. Routing protocols are based on distributed algorithms that enable minimum cost paths (*i.e.,* the set of hops between nodes) through a network to be established. Examples of common routing protocols include RIP (Routing Information Protocol), which is a distance vector protocol based on the Bellman-Ford algorithm, or OSPF (Open Shortest Path First), which is a link state protocol based on Dijkstra's algorithm. While both of these algorithms result in shortest path forwarding tables, OSPF is the more robust of the two and is the most widely used intra-domain routing protocol in the Internet today.

It is important to note that there is a different routing protocol that is used to establish paths *between* networks (administrative domains). This protocol is called Border Gateway Protocol (BGP), and is the subject of Lab #5.

1 Overview and Objectives

While the basic conceptual aspects of configuring and managing routers in a single administrative domain are simple, the process becomes extremely complex as the size and diversity of a network increases. Imagine a simple network with, say less than 10 routers that is only concerned with forwarding packets along the shortest paths. In this case, configuration is quite simple. Conversely, imagine a network like the large service provider that includes tens of thousands of routers spread out all over the world, connected to hundreds of other networks and offering services like private networking, voice and video over IP, etc. to its customers. In this case, the tasks of maintaining, configuring, and troubleshooting become immensely complicated. While examining the issues of scale and diversity are beyond the course, the exercised in this lab will provide a starting point for understanding those issues.

Lab 4 is divided into two parts. The first part will introduce you to the layer 3 configuration and routing capabilities on Linux hosts. While end host PC systems are never used as routers in live, high-speed networks, understanding their routing capabilities is important for experiments with real routers in WAIL. It is important to note that PC-based routers are also useful for conducting routing experiments and tests since commercial routers are closed systems (*i.e.,* you only have command line access and no ability to load and run your own code on them).

The second part of this lab will give you some experience with configuration and management of real routers. The devices that are available for this lab are so-called access system *i.e.,* systems that would typically be attached to local area networks as the first layer 3 hop on an end-to-end path through the

Internet. These devices are typically small in the sense that they have a limited number of relatively low speed ports (*e.g.,* gigabit Ethernet), but they are often feature-rich (*e.g.,* MPLS, VoIP, access control, etc.) devices that provide a wide range of configuration options. Examples of systems that you might use in this part of the lab include the Cisco 7500 or the Cisco 7300.

Upon completion of the lab exercises below, students will be able to:

1. *Understand the basic aspects of Cisco IOS command line interface.*
2. *Understand the basic steps involved in setting up static routes between nodes.*
3. *Understand the basic operation of the OSPF intra-domain routing protocol.*
4. *Setup an administrative domain that uses OSPF for intra-domain routing*
5. *Troubleshoot routing problems in an administrative domain.*

2 Part 1: Layer 3 Capabilities on End Hosts

Communication between hosts that are not located in the same local area network (or in LANs connected via a bridge) must take place using layer 3. In other words, the packets must encounter a device that will forward packets using the destination address included in the IP header of packets transmitted by the sending host. In fact, all packet transmitted by standard applications running on end hosts include an IP header. This is done by the networking component of the operating system running on the host.

Most layer 3 networking on end hosts takes place through static routes *i.e.,* routes that are established by hand and are not subject to change by participation in dynamic routing protocols such as RIP or OSPF. These routes enable packets with specific destination IP addresses to be directed to specific network interfaces on the system. Static routes entered on end hosts may no longer be valid when a machine is rebooted. While static routes are infeasible for large networks, they are used on end hosts or in small networks that are stable and not typically subject to link failures caused by external events (such as a back-hoe inadvertently cutting a link). Static routes are also used by network administrators to establish specific links between nodes that are meant to be permanent, such as peering links between administrative domains.

2.1 The *route* command

Like most functions on end hosts, there are several command line interfaces that enable routing functions to be monitored and configured. The **netstat -rn** (-r is for routing table -n designates that the output should be IP addresses and not host names) can be used to display the routing table information on an end host. Netstat is a standard interface available on Unix systems.

The **route** command allows an administrator to view, add, and delete *static routes*. The route command is often used in conjunction with the ifconfig command to set up all networking on end hosts. It is important to note that networks have *netmasks* associated with them. A netmask defines which portion of an IP address identifies the network and which portion identifies the host. For example, if the first 24 bits of the IP address designate the network and 8 bits correspond to the host, the netmask is 255.255.255.0. Please see the pre-lab readings for more information on netmasks. If you run the **route** command you will notice that there is a *gateway* associated with each entry. The gateway acts as an intermediary network host for that particular route between the source and the destination. A Linux PC set up to forward packets can act as 5 a gateway. Standard uses of the **route** command include:

- *route -n* displays the routing table with IP addresses instead of host names.
- *route add -net netaddress netmask mask gw gatewayaddress* where example values could be netaddress = 192.168.13.0 mask = 255.255.255.0 gateway = 192.168.14.1
- *route add -host hostaddress gw gatewayaddress* is used when you want to specify a route to a specific host.
- *route del -net netaddress netmask mask gw gatewayaddress* removes the specified entry of the routing table. Note that it is not necessary to provide all arguments as long as the set provided represents a unique entry.

Linux PC's typically already have the kernel support necessary to forward packets. A typical end host just has a default route set to a first hop gateway. The only difference between this typical PC and a linux PC routing betweeen multiple networks is the number of rules in the routing table, which are easily accessible via the route command. However, by default the Linux PC does not forward any packets received by its interfaces. This behaviour is controlled by the sysctl variable net.ipv4.ip_forward. To change this behavior on a running system you can use the sysctl utility, read them man page. To have this behaviour automatically set on boot you would have to change a configuration file. Which configuration file, is distribution dependent, but the FORWARD_IPV4 option in /etc/sysconfig/network for RedHat based distro's, or /etc/sysctl.conf for most others is a good place to start. In Schooner the system images already have the option set to forward packets so you won't have to change anything in your experiments.

2.2 Open Shortest Path First Routing

Dynamic routing protocols enable networks to adapt to failures and outages. As mentioned above, the two most widely used dynamic routing protocols are OSPF and RIP. Both of these protocols are used to establish lowest cost routes in intra-domain environments. OSPF is the more widely used of the two since it converges faster and avoids the count to infinity problem.

OSPF uses *reliable flooding* as the basis for propagating link state information throughout a network. All routers broadcast link information to all other routers (not just neighbors as is done in RIP). From this information individual routers can piece together the topology of the network and using Dijkstra's algorithm, establishing forwarding tables that result in lowest cost paths being followed. In extremely large networks, there is a significant cost for flooding all routers with link state information, and the processing cost of piecing together paths can be extremely prohibitive. However, there are various ways to mitigate these costs.

In order to gain familiarity with OSPF you will be setting up a small network consisting of Linux PC's configured to act as OSPF capable routers.

2.3 Quagga, the new Zebra

While routing protocols are not built into base Linux distributions, a number of packages are available for dynamic routing. One of the most popular of these is called Quagga, a fork of the older Zebra package. As you have leared the Linux kernel internally supports packet forwarding and routing via static routes. The dynamic routing capabilities of the Quagga package just change the the forwarding tables of the kernel.

The Quagga routing package operates as a number of system daemons. The first daemon is named "zebra", this daemon takes care of global issues but primarilly is responsible for taking information from other daemons, and updating the kernel routing tables with the appropriate routes. The other daemons, are instances of a particular routing protocol, RIP,RIPng,OSPF,and BGP are all supported.

2.4 Experimental Setup

The topology for the first part of Lab4 is shown in the prelab. Nodes that will run dynamic routing daemons will need the Quagga RPM installed, and a script run at boot which fixes a number of configuration issues to allow the daemons to run. Use the link from the prelab to an appropriate NS file for you experiment.

The first two tasks will use static routing, but for the last task you will setup Quagga to do dynamic routing on all of your nodes. In this topology are configurations are relatively simple. First you will have to login to each node and start the zebra and ospf processes.

```
[blodge@node1 ~]$ sudo /etc/rc.d/init.d/zebra start
Starting zebra: Nothing to flush.
                                                           [  OK  ]
[blodge@node1 ~]$ sudo /etc/rc.d/init.d/ospfd start
Starting ospfd:                                            [  OK  ]
```

After you have started both proccesses you need to connect to the CLI for your quagga router. With the way Quagga is structured, there are actually multiplie CLI's so pay close attention, the OSPF router console is on port 2604.

```
[blodge@node1 ~]$ telnet localhost 2604
Trying 127.0.0.1...
'autologin': unknown argument ('unset ?' for help).
Connected to localhost.
Escape character is '^]'.

Hello, this is Quagga (version 0.98.6).
Copyright 1996-2005 Kunihiro Ishiguro, et al.


User Access Verification

Password:
ospfd> en
ospfd#
```

The password is "zebra" and do an "en" for enable right away. You will find that the Quagga CLI is similar to the IOS work you have already done, as was mentioned in class, most network gear is pretty similar to configure, the exact syntax might differ a bit, but you usually can wander around a device and find what you need.

```
ospfd# config t
ospfd(config)# router ospf
ospfd(config-router)# router
ospfd(config-router)# router-id 10.0.0.1
ospfd(config-router)# network 198.168.0.0/16 area 1
ospfd(config-router)# no passive-interface eth0
ospfd(config-router)# end
ospfd#
```

So from the CLI, we enter configuration mode, and go to the ospf context. You might ask, why do we have to go into a different context if this is the OSPF CLI and only does OSPF related configuration, well most routers don't have different CLI's to telnet into like Quagga, you'll see later on the Cisco, from the regular CLI, you can configure the OSPF capabilites.

OSPF uses a router-id variable to identify routers, depending on the platform it might choose one for you, usually via some list of rules, which depending on what you change it the router might fluctuate, so it's best to nail it down, it doesn't matter what you use, just that they be unique. The 'network' line tells OSPF that it should announce routes for and try to form OSPF adjancies for anything in the 198.168.0.0/16 IP space, and then we turn on OSPF on the physcial interface eth0. Depending on which node in Schooner you get, you will have to use different physical interfaces, once the experiment is booted identify your control interfaces, you don't want to talk OSPF out them, or you will be bypassing your experimental topology.

2.5 Part 1 Tasks

1. Swapin the topology, and configure static routes on all the simulated routers. For this task don't change the default route, add a route entry for each subnet. It is best to figure out the commands on paper, and then login to each machine and run the commands. Verify connectivity between the simulated routers by logging into each one, and pinging the others.

2. Now simulate a link failure, on one of your simulated routers down an interface with "ifconfig <interface> down". What commands do you need to run to reestablish connectivity between nodes? How long does it take you to figure it out, and how long does it take you to actually login to the machines and run the commands?

3. Swapout and swapin your topology again. Setup OSPF and verify your nodes are getting routes via 'route -n' for non locally connected networks. Connect to all the nodes and verify connectivity like you did in Task 1. Now down an interface again, what commands do you need to establish connectivity now?

3 Part 2: Intra-Domain Networking with Routers

While PC-based routers are useful for understanding how layer 3 networking and router work, they are never used in high speed environments or environments that require a large number of link interfaces. Routers are devices that power the Internet. They provide both control and data plane capability along with a host of features and interfaces. To address the requirement of high availability, routers are also designed with redundancy (*e.g.,* multiple power supplies) to enable them to function with high reliability. Like PC's, routers run an operating system that controls the underlying hardware, enabling it to be configured and monitored.

3.1 Cisco Internetworking Operating System

In order to interact with a Cisco Router or network switch you will have to become familiar with the Cisco Internetwork Operating System (IOS). Cisco IOS has components that control both routing and switching and are presented as a multitasking operating system. Please go through the tutorial listed in the pre-lab to familiarize yourself with IOS. If you wish to login to a router there is a project currently running on Schooner that will allow you to login and try different commands. See the below URL for details.

http://www.cs.wisc.edu/p̃b/640/PA1/routerTutorial.htm

Note that router configurations in WAIL are fixed for the most part. This is because there is limited ability to reconfigure the physical links that interconnect systems. In experiments that only include PC's, the physical interconnections can be faked with Virtual LANs. You can access the router-based configurations that are available in WAIL by accessing the "scenarios" link on the left side of the Schooner interface.

The Cisco IOS command line interface can run in different modes which correspond to levels of privilege. There are fifteen different priviledge levels, but the two you will hear refered to commonly are "user" and "privileged", which usually correspond to level one and level fifteen.

To enter level 15 or privileged mode it is necessary to run the command **enable** and provide the correct password. If you want to know the commands which are available use **?** to prompt for help. The following list gives commands which are commonly used (in privileged mode). Note that IOS allows unambiguous abbreviated words to substitute for commands e.g., "conf t" produces the same result as "configure terminal". Examples of other IOS commands are as follows:

- *show running config* Provides router configuration information
- *reload* Reboots the operating system
- *show protocols* Gives protocol information
- *show ip route* Gives routing table
- *show interfaces* Gives interface information

- *configure terminal* Enters global configuration mode
- *interface [interface]* the interface could be something like Ethernet0/0, run under the global configuration mode

3.2 Static Routes in IOS

Cisco Routers also allow static routes. Issue the following commands:

- *conf t* Enters the global configuration mode
- *ip route dest mask gatewayaddress* Adds a static route to dest via gatewayaddress which can potentially be 0.0.0.0

To remove the entry simply add "no before the "ip route" command.

3.3 OSPF in IOS

In order to get OSPF working on Cisco routers you will need to change the default configuration. First you must enable an OSPF process that will manage the link state information. Next, you must set the network which the OSPF process will be applied to. In addition, be sure that the IP addresses of your router interfaces are correctly set. The commands are listed below:

- *enable* Enter privileged mode
- *configure terminal* Enter global configuration
- *no router rip* Disable RIP (on by default)
- *router ospf 1* Initialize an OSPF process with ID 1, don't use 0 as it is reserved
- *network 192.168.13.0 0.0.0.255 area 1* Associate the OSPF process with traffic from a particular network
- *interface Ethernet0/1* Set the IP Address for your interfaces (optional)
- *ip address 192.168.13.2 255.255.255.0* Address then mask
- *interface Ethernet0/2*Set the IP Address for your interfaces (optional)
- *ip address 192.168.14.2 255.255.255.0*Address then mask
- *end* Exit Interface configuration
- *clear ip route ** Reset all of the OSPF processes on the router

4 Tasks

1. Run through the Cisco IOS tutorial and be sure that you know how to change IP addresses, enter all configuration modes, set/view routes, restart the router, view the current configuration, and view access lists.
2. Configure the PC's and routers in Topology 2 to use OSPF. Upon completion all host PC's should be able to communicate.
3. Use traceroute between node0 and node2, compare the traceroutes running in different directions.