CS 638 Lab 5: Inter-Domain Routing

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

The Internet is a network of networks that forward packets based on their destination IP addresses. Routers are the devices responsible for packet forwarding. Routers use routing protocols to identify lowest cost paths between networks (ranges of IP addresses) and thereby establish their forwarding tables. In Lab #4 we learned about how paths were established within a single administrative domain *i.e.,* intra-domain routing. However, the protocols used for intra-domain routing are separate and distinct from the protocol used to establish routes *between* administrative domains.

Interconnections between administrative domains are defined by business relationships. An individual administrative domain, also referred to an an Autonomous System (AS), is an entity that owns at least one IP address range and an AS number (both assigned by ICANN). Note that an administrative entity can also own/operate more than one AS. AS's must have physical connection to other AS's in order to communicate – the set of interconnected AS's comprises the Internet. Small or medium size AS's connect to upstream providers at local, regional or national Points of Presence (POPs). When an AS pays another AS to connect, this is called a Customer-Provider business relationship. AS's that are approximately the same size or that have some other interest in each other can also have non-paying relationships. In this case, their interconnections are called peering relationships (explained in greater detail a bit later). In either case (customer-provider or peering), in addition to the physical interconnections, the AS's must establish communications channels between each other that enable information about the networks that are available from each to be sent and received.

The protocol that is use to send and receive information about the networks that are available between administrative entities is called Border Gateway Protocol (BGP). BGP is the only protocol used for this task, and in that sense it play an *extremely* important role in the Internet. BGP is concerned with establishing loop-free routes between autonomous systems. To accomplish this BGP uses *path vector exchanges.* Path vectors begin with a network address range expressed using CIDR slash notation (*e.g.,* 192.18.0.0/16 which specifies a class B network beginning with the address 192.18.0.0) and a set of AS numbers which if followed will conclude with the AS that owns that address range. As successive AS's receive path vectors, they append their own AS number and then propagate the information to any other AS to which they have a BGP session. If they see their own AS number in a path vector that they receive, they do not propagate it further and in that way prevent routing loops.

BGP sessions are established between BGP speakers, which are routers designated for this purpose by an AS. There is only one BGP speaker per AS, however there can be and often are more than one point at which two networks physically interconnect. If there are multiple interconnection points between networks, then it is up to the network administrators to determine where and how to transmit packets between the networks. These are often complicated decision that depend on the business objective of both entities (and sometimes others!).

In addition to the path vectors themselves, certain other attribute information can be transmitted in updates. One of these, Multi-Exit Discriminators (MEDs) are meant to be used to express local information to neighboring AS's. While this idea makes good conceptual sense, the lack of standards for MEDs often renders them useless. This is just one of many examples of the shortcomings of BGP. Despite these problems, packets still get transmitted successfully most of the time!

1 Overview and Objectives

Similar to lab #4, configuring and managing an interconnection to an upstream provider for a single small AS is simple. However, for large service providers that provide upstream connections to hundreds of customer AS's and that have peering relations with many others, managing BGP and all of the associated physical connects is extremely complex. Like intra-domain routing in large networks, there are issues of the dynamics in the physical environment (*e.g.,* links going down or coming up), and also with the fact that there are humans in the loop who can cause problems. Imagine, for example, that the operator of a small AS accidentally types in the wrong address range for his network advertises to his upstream provider that he now owns all of AT&T's address space! If this route were propagated, it could cause huge swings of traffic. Things like this have happened and illustrate the care and attention that is required by large ISP's that operate the core backbone components of the Internet.

As noted in the introduction and pre-lab, BGP actually has two components - EGRP and IGRP, which handle the external and internal aspects of BGP respectively. This lab focuses on external BGP. eBGP is a way for network administrators to set policies for their autonomous system. Policies are typically based on financial constraints. For example: Large Company A is paying us lots of money to act as their service provider. We agree to forward their traffic, so we announce the availability of Large Company A's subnet to all of our neighbors. Based on our neighbors' BGP policies they will decide if they will continue to propagate the message or not.

You will set up a network of nodes that are meant to represent a set of AS's that are physically interconnected. Each node in this lab is meant to represent an entire AS and will act as that AS's BGP speaker. In a live deployment, the BGP speaker would inform the other routers in its AS of routing paths using an Interior Gateway Routing Protocol (IGRP). In this lab, in order to create networks that have a slightly larger size, a combination of PC's and real routers will be used to create the network. In order to do this, specialized software must be used on the PC's to enable them to speak BGP to other nodes. The objective of this lab is to gain experience with BGP configuration on both PCs and routers and to observed how BGP communication takes place between neighbor networks.

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

- *Understand BGP message propagation*
- *BGP configuration on a Cisco routing and on a Linux PC*
- *Understand Customer/Provider and peering relationships*

2 Setup

Download the NS file from the pre-lab page and begin an experiment. The primary file uses both hardware routers and PC running Quagga to build our topology, while the alternate configuration uses just PC nodes if you want to test something or there is a shortage of Dumbbell scenario instances. You will probably want to make a few hard copies of the topology to write notes on and keep track of which configuration you've changed. Submit your NS file, and swap it in.

Schooner allows a hardware router to be in use by multiple experiments at one time. The router scenarios available in Schooner use a fixed addressing scheme, which allows this multiplexing. As a simple security precaution and to prevent packets from flowing between scenarios, access control lists are used to police packets. For this lab we will declare the router to be allocated exclusively to your experiment, and will manually remove the access control lists. If you are working with the alternate NS configuration you wouldn't have to compete this steps.

Login to each router that has been allocated to your experiment. Examine the existing configuration on the interfaces and then in configuration mode remove the access control lists from the interface.

```
> telnet r-c7300-32-i
Trying 10.0.1.52...
Won't send login name and/or authentication information.
Connected to r-c7300-32-i.schooner.wail.wisc.edu.
Escape character is '^]'.

User Access Verification

Username: wail
Password:

r-c7300-32-i#show run int gigabitEthernet 0/1
Building configuration...

Current configuration : 301 bytes
!
interface GigabitEthernet0/1
 description access(r-c6500-12-b,Gi6/9)
 ip address 198.18.0.17 255.255.255.240
 ip access-group Dumbbell_2 in
 ip access-group Dumbbell_2 out
 ip route-cache flow
 no ip mroute-cache
 duplex auto
 speed auto
 media-type rj45
 no negotiation auto
 hold-queue 30 in
end

r-c7300-32-i#config t
Enter configuration commands, one per line.  End with CNTL/Z.
r-c7300-32-i(config)#int GigabitEthernet0/1
r-c7300-32-i(config-if)#no  ip access-group Dumbbell_2 in
r-c7300-32-i(config-if)#no  ip access-group Dumbbell_2 out
r-c7300-32-i(config-if)#end
r-c7300-32-i#write mem
Building configuration...
[OK]
r-c7300-32-i#
```

2.1 Daemons

Since it is not possible to give every group 7 routers, we will be using nodes that behave as routers. Again we can use the Quagga software daemon to have our nodes interact via a routing protocol.We will be using two of the routing daemons: zebra and bgpd. Similar to Lab 4, the Zebra daemon manages the individual routing protocol daemons, and interacts with the forwarding tables that exist in the Linux kernel. The BGPd daemon will allow nodes to behave as a router with BGP enabled. As with prior labs, the FC6-STD Operating System image must be used.

The configuration files for the daemons can be found in the directory /etc/quagga/. The default login and enable password is zebra. These files contain the startup configuration files for the daemon, which does not necessarily match the existing running configuration. Just like the Cisco command line you can use "write mem" to save the running configuration to the startup configuration. In the NS file you might notice the nodes run a script at startup, this takes care of copying some default configurations, and adding some local user accounts required to run Quagga. Feel free to make your own copy of the script, and add to it, just remember to change your NS file. Automation is a key component of designing and maintaining a modern network.

The BGPd daemon on startup acquires information about the local interfaces from the Zebra daemon, so make sure you start it first, and then start BGPd with the default configuration files.

```
[blodge@node3 ~]$ sudo /etc/rc.d/init.d/zebra start
Starting zebra: Nothing to flush.
                                                           [  OK  ]
[blodge@node3 ~]$ sudo /etc/rc.d/init.d/bgpd start
Starting bgpd:                                             [  OK  ]
```

The CLI for the Zebra daemon is running on 2601, and BGPd on 2605

```
[blodge@node3 ~]$ telnet localhost 2605
Trying 127.0.0.1...
'autologin': unknown argument ('unset ?' for help).
Connected to localhost.
Escape character is '^]'.

Hello, this is Quagga (version 0.99.7).
Copyright 1996-2005 Kunihiro Ishiguro, et al.

User Access Verification

Password:
bgpd> en
bgpd#
```

2.2 BGP Configuration

Connect to the BGP host and enter configuration mode. Type "router bgp asn" to enter the area where you can issue bgp commands (note: asn should be specified in the configuration file). As always show commands will not work while in configuration mode.

See the Quagga page for a more comprehensive summary of BGP commands.

Basic BGP commands include:

- *router bgp asn* Sets the asn of the router and enters bgp command mode
- *no router bgp asn* Removes the asn of the router

- *neighbor x.x.x.x remote-as asn* Sets the asn specified as a neighbor and the IP specified as the BGP speaker for that AS
- *no neighbor x.x.x.x remote-as asn* Removes the neighbor
- *network x.x.x.x/y* Announces the subnet specified
- *no network x.x.x.x/y* Removes the previously announced subnet
- *show bgp neighbors* Lists the neighbors
- *show ip bgp* Shows BGP routes
- *show ip bgp summary* Shows general BGP information

Start by configuring node1 and node2. Steps for doing this are as follows (assuming all interfaces are properly configured):

- Edit /etc/quagga/bgpd.conf on node1 and node2 to set the correct ASN
- Start zebra and bgpd on both nodes
- Connect to bgpd on node1 and set AS 121 as a neighbor
- Connect to bgpd on node2 and set AS 12 as a neighbor
- Use the network command to announce a fictitious subnet from node2
- After a few seconds show the bgp routes on node1 to verify that the new subnet was added
- Use no network to remove the invalid subnet

Repeat this to configure the entire network properly. Once every node is configured, announce all of the subnets in the network.

2.3 Initial Tasks

- Verify you have full connectivity using ping between every pair of nodes.
- Using the BGP tables and traceroute confirm, the path from node0 to node3
- On R1, shutdown the peering to R2, use the "neighbor peer shutdown" command, to temporarily shutdown the peering without removing your configuration
- Now confirm the path from node0 to node3 again

Remove all the network announcements before moving onto the next section.

3 Customer/Provider Relationships

As explained above, a customer/provider relationship is defined by the business relationship between two parties (specified in a Service Level Agreement). A larger entity (the provider) will agree to carry a certain amount of traffic (to and from) for a smaller entity (the customer) at a specified price (typically dollars per unit volume) and a specified level of quality (availability, loss, jitter, etc.). This creates a layered view of the Internet. Very large backbone providers provide connectivity and bandwidth to large corporations. The large corporations then act as a provider to medium sized corporations, and so on. Entities can have multiple providers, as well as multiple customers.

The cost of bookkeeping bandwidth usage can become significant. This along with various other reasons motivates *peering* relationships. A peering relationship is formed between two entities of about the same size that agree to carry traffic for each other without charge. Two entities that transfer approximately equal amounts of data through the others network have incentive to carry the each others traffic free of cost to avoid the overhead of traffic monitoring and other bookkeeping. Take the case where the entities did decide to charge each other, each company would charge the other approximately equal amounts plus the bookkeeping costs. By creating a peering relationship the bookkeeping costs are avoided.

There various way in which network interconnects can be made result in the possibility that of multiple paths for some routes in the network. The traffic an AS is asked to forward is determined by what BGP announcements that AS forwards. Choosing not to forward a BGP announce message prevents anyone else from learning of your path to the announced subnet. Routes can also be aggregated (using CIDR) which has the benefit of keeping BGP routing table at a more manageable size. There are many ways in IOS you can specify forwarding, but you should be able to accomplish everything needed in this project with only two: ip as-path access-list and neighbor distribute-list. Your task is to setup BGP forwarding rules so that the network is setup as follows:

- AS 12 could potentially be asked to forward traffic from AS 1. AS 1 is AS 12's provider, so it is assumed that AS 1 is receiving payments from node1 and also AS 1 has the larger faster network of the two. AS 12 does not desire to carry AS 1's traffic bound for AS 21.
- Similarly, AS 21 should refuse to carry AS 2's traffic bound for AS 12
- AS 12 is a moderate sized network and would like to keep its forwarding of traffic to a minimum. AS 12 refuses to carry AS 21's traffic bound for AS 1
- Similarly, AS 21 should refuse to carry AS 12's traffic bound for AS 2
- To avoid the cost of sending traffic through the provider AS 12 should route all traffic bound for AS 21 or AS 211 through the peer link
- Similarly, AS 21 should route all traffic bound for AS 12 and AS 121 through the peer link

Announce all the subnets once again. Verify all of the routes follow the rules above.

4 Customer/Provider Tasks

1. Modify your BGP configuration to correspond Consumer/Provider peering relationship as described
2. Verify connectivity with ping and traceroute
3. Repeating your earlier experiment, down the peering relationship between R1 and R2, does traffic automatically route around the problem? Explain what has happened and why?

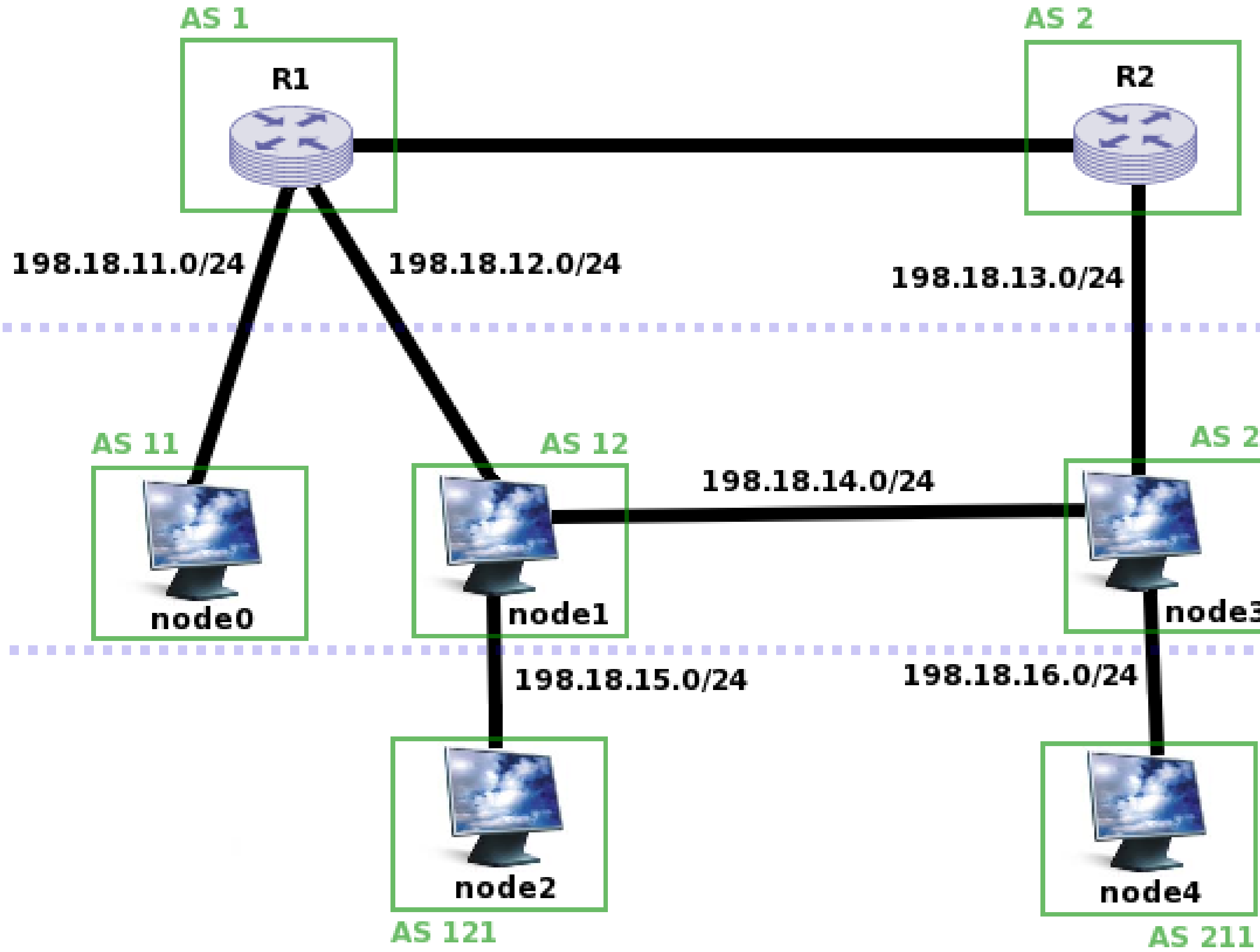

Figure 1: Network Topology