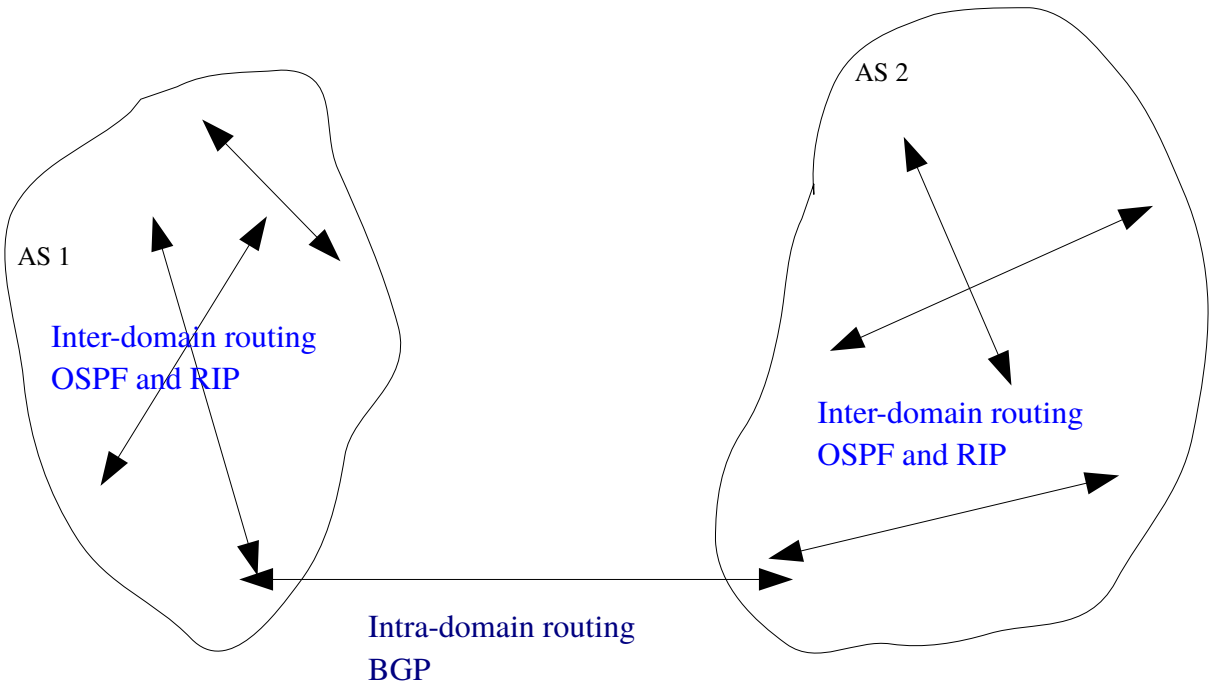
Rudiments of Routing

Moving bits from the source to the destination is a major function of computer networking. On the current Internet, the Network layer is responsible for achieving this.



In general, most routing within Autonomous Systems use Routing Information Protocol (RIP), or its enhanced version Open Shortest Path First (OSPF). The current de facto Intra-domain routing standard is Border Gateway Protocol(BGP),Version 4 (RFC 4271) . You need to concern yourself with these protocols if you are dealing with routers inside or between Autonomous Systems.

Most currently implemented routing protocols between routers use either the Link-State algorithm, which is based on the famous Dijkstra's algorithm in Graph Theory, or the Distance-Vector algorithm, based on the Bellman-Ford equation.

At the host level, however, most likely you need only a static routing table. This is a table of routes that the OS kernel keeps. It is possible to add to and delete from routes in the kernel routing table relatively easily. We discuss routing tables based on RIP (RFC2453).

The routing tables are operating-system dependent. On linux-based systems, the netstat command with a -r flag will give the routing table. On Windows machines, route print on a command prompt will show the routing table. Below are two linux-OS outing tables and a Windows 10 routing table.

```
fobn5:~/CS2/Week10$ netstat -rn
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         137.140.8.250   0.0.0.0         UG        0 0          0 enp0s25
137.140.8.0     0.0.0.0         255.255.255.0   U         0 0          0 enp0s25
fobn5:~/CS2/Week10$ uname -a
Linux fobn5 4.12.14-300.fc26.x86_64 #1 SMP Wed Sep 20 16:28:07 UTC 2017 x86_64 x86_64 x86_64 GNU/Linux
fobn5:~/CS2/Week10$
```

```
wyvern:~$ netstat -rna
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
0.0.0.0         137.140.4.250   0.0.0.0         UG        0 0          0 eth0
137.140.4.0     0.0.0.0         255.255.255.0   U         0 0          0 eth0
wyvern:~$ uname -a
Linux wyvern.acsl.newpaltz.edu 3.10.0-693.el7.x86_64 #1 SMP Thu Jul 6 19:56:57 EDT 2017 x86_64 x86_64 x86_64 GNU/Linux
wyvern:~$
```

```
easwaran@LAPTOP-UGUQHM52 ~
$ route print
===========================================================================
Interface List
 17...94 65 9c d3 54 33 ......Microsoft Wi-Fi Direct Virtual Adapter
 18...94 65 9c d3 54 32 ......Intel(R) Dual Band Wireless-AC 7265
  9...94 65 9c d3 54 36 ......Bluetooth Device (Personal Area Network)
  1...........................Software Loopback Interface 1
  8...00 00 00 00 00 00 00 e0 Microsoft Teredo Tunneling Adapter
 11...00 00 00 00 00 00 00 e0 Microsoft ISATAP Adapter #3
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0    137.140.168.1  137.140.168.165     40
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
    137.140.168.0    255.255.248.0         On-link   137.140.168.165    296
  137.140.168.165  255.255.255.255         On-link   137.140.168.165    296
  137.140.175.255  255.255.255.255         On-link   137.140.168.165    296
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link   137.140.168.165    296
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link   137.140.168.165    296
===========================================================================
Persistent Routes:
  None

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  8    331 ::/0                     On-link
  1    331 ::1/128                  On-link
```

When looking at routing tables, remember that most Unix-like operating systems use mnemonic names for their interfaces. In the first two images of routing tables above, the Iface column has names enp0s25 and eth0 respectively. These are the mnemonic names of the network interfaces attached to the hosts.

You can see all the configured interfaces on a linux host using the ***ifconfig*** command which is usually found in */sbin/* directory (but not always). On Windows machines, you can run the "ipconfig /all" command in a command terminal to see all the configured interfaces (plus other information). On both Linux and Windows (I think!), you can see the routing table with “netstat -r” command.

The output from ***/sbin/ifconfig*** command on a linux machine below shows that there are four configured interfaces, one an Ethernet (enp6s0), another for a docker instance(docker0) and a VPN tunnel (cscotun0) interface, and the loopback interface (lo).

```
linux5:~/Videos$ /sbin/ifconfig
cscotun0: flags=4305<UP,POINTOPOINT,RUNNING,NOARP,MULTICAST>  mtu 1350
        inet 137.140.237.184  netmask 255.255.254.0  destination 137.140.237.184
        inet6 fe80::f4de:c6ee:c784:7970  prefixlen 64  scopeid 0x20<link>
        inet6 fe80::2ff1:6847:7848:49e5  prefixlen 128  scopeid 0x20<link>
        unspec 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00  txqueuelen 500  (UNSPEC)
        RX packets 13301  bytes 3541177 (3.3 MiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 13661  bytes 2089113 (1.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 0.0.0.0
        ether 02:42:9b:7c:2a:fd  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp6s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.2.103  netmask 255.255.255.0  broadcast 192.168.2.255
        inet6 fe80::be30:5bff:feb8:3947  prefixlen 64  scopeid 0x20<link>
        ether bc:30:5b:b8:39:47  txqueuelen 1000  (Ethernet)
        RX packets 2858675  bytes 1396176062 (1.3 GiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 1929370  bytes 353320570 (336.9 MiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
        device interrupt 17

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1  (Local Loopback)
        RX packets 2012  bytes 325125 (317.5 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 2012  bytes 325125 (317.5 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The real physical interface in the above figure is the ethernet interface – the others are conceptual devices that use the ethernet. We have already seen the loopback interface before.

In the following, we generally omit loopback interfaces for convenience.

Let us look at the routing table on a linux host carefully:

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	virbr0
137.140.6.0	0.0.0.0	255.255.254.0	U	0	0	0	eth0
169.254.0.0	0.0.0.0	255.255.0.0	U	0	0	0	eth0
0.0.0.0	137.140.7.250	0.0.0.0	UG	0	0	0	eth0

The first line says that the network 192.168.122.0/24 can be reached by directly sending to virb0 interface - this is how we reach virtual machines on this host. The second line says that 137.140.6.0/23 can be reached directly sending to eth0 interface. The flag U here means that the interface is “UP”. We know the netmask is 23 because of the 255.255.254.0 entry under Genmask.

The third line, 169.254.0.0/16 is a special class of IP link-local addresses using the draft Zeroconf standard. Zeroconf is better known in its incarnation as Apple's Rendezvous. Routes like this one are stop-gap responses that allow Linux to somewhat participate in Zeroconf/Rendezvous networks. For more information, look at zeroconf.org.

The fourth line is the catch-all route. It says that if the above three lines fail to give a route, send the packet to 137.140.7.250. This is called the default route and the gateway machine is called the default router. The UG flags mean that the route is UP and that the destination is a network not directly connected to this interface. Note that from the first line we already know how to get to 137.140.7.250 – simply send to eth0 interface. 137.140.7.250 must be directly on the wire.

Routing table flags.

Usually, in most simple host routing tables on linux/Unix hosts, one sees three sets of flags - U, G and H. U means route is up, G means the destination is a network not directly connected to us, and H means the destination is a host IP. So for example, a routing table entry with flags UGH means the the destination in the entry is a host IP on a network not directly attached to us. UH flags mean that the destination is a host IP directly attached to us.

Routing table – another example

Consider another example:

Kernel Routing Table							
Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
192.168.5.20	192.168.10.7	255.255.255.255	UGH	1	0	180	eth1
192.168.1.81	192.168.10.5	255.255.255.255	UGH	1	0	187	eth1
192.168.10.0	0.0.0.0	255.255.255.0	U	0	0	63311	eth1
192.168.18.0	0.0.0.0	255.255.254.0	U	0	0	753430	eth0
192.168.64.0	192.168.10.5	255.255.192.0	UG	1	0	47543	eth1
192.168.128.0	192.168.10.7	255.255.192.0	UG	1	0	89011	eth1
127.0.0.0	0.0.0.0	255.0.0.0	U	0	0	564	lo
0.0.0.0	192.168.10.20	0.0.0.0	UG	1	0	183436	eth1

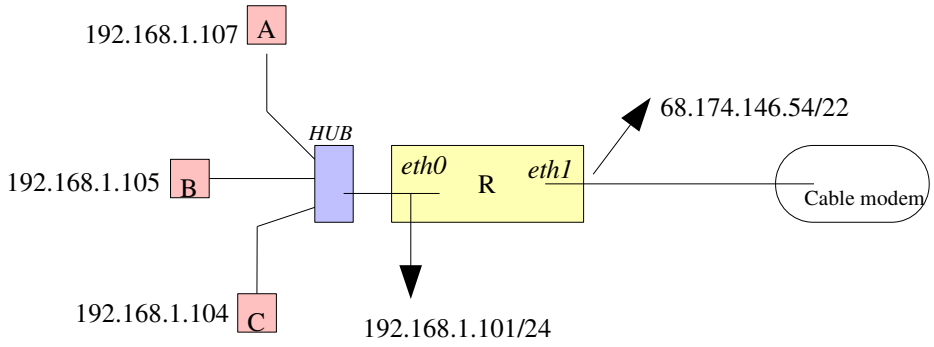
Let's interpret the line highlighted in blue. It says that if the destination of a datagram is 192.168.64.0/18, then send it to 192.168.10.5. In other words, if the first 18 bits of a destination IP matches that of 192.168.64.0 (that is: 11000000 10101000 01), then send it to 192.168.10.5. For instance, if this host wants to send a datagram to 192.168.83.105, the datagram should be routed to 192.168.10.5, because the first 18 bits of 192.168.83.105 matches this rule.

Notice how *genmask* (or *netmask)* is used in the above interpretation. Look at the line highlighted in yellow. This rule says that if all 32 bits match those of 192.168.1.81, send to 192.168.10.5 - in other words, if you want to send a datagram to the IP 192.168.1.81, send to 192.168.10.5.

As an exercise, try to construct the network topology attached to the router whose routing table is given above.

Another example.

Consider a scenario as follows. You have a machine R with two Ethernet network cards eth0 and eth1; one is hooked up to your cable modem as shown, and the other interface is connected to a hub to which hosts A, B and C also connect.



This is a set up you can use if you do not have a commercial Cable/DSL router. In fact, home routers are simple machines like *R* in the figure above, running linux and a web server for system administration.

What will be the routing table on R? It should specify that all packets to 192.168.1.0/24 should be sent to eth0 interface. All packets to 68.174.144.0/22 network should be sent to eth1 interface (explain this!). Everything else should be sent to a default gateway that your ISP specifies, say 68.174.144.22. The routing table looks somewhat like this:

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
68.174.144.0	0.0.0.0	255.255.252.0	U	0	0	0	Eth1
0.0.0.0	68.174.144.22	0.0.0.0	UG	0	0	0	Eth1

The IP, netmask, default gateway, as well as domain name servers are specified by your ISP, mostly through DHCP (Dynamic Host Control) protocol.

What would the routing table of a host like A look like? It's quite simple:

Destination	Gateway	Genmask	Flags	MSS	Window	irtt	Iface
192.168.1.0	0.0.0.0	255.255.255.0	U	0	0	0	eth0
0.0.0.0	192.168.1.101	0.0.0.0	UG	0	0	0	eth0

This says that to send to 192.168.1.0/24 network, just send to the eth0 interface, send everything else to 192.168.1.101 (which we know how to get to from the previous line).

Within R, eth0 and eth1 interfaces should be configured to forward packages to each other.

If A in the above diagram needs to send a datagram to 137.140.8.102, A's IP will look up the routing table, and see that the datagram should b sent to its default gateway, 192.168.1.101. Note that this does not mean that the destination IP in the IP header will be 192.168.1.101 – that will still be 137.140.8.102. That the immediate next hop is 192.168.1.101 is relevant only to the link layer, as we will see soon.

Bringing up an interface (the following applies only to linux/unix hosts).

On most Unix/Linux machines, the ifconfig command can be used to bring up and shutdown network interfaces. For example, the following command brings up the eth1 interface and assigns it an IP:

root>/sbin/ifconfig eth1 192.168.1.105 netmask 255.255.255.0 up

Similarly you can shutdown an interface by

root>/sbin/ifconfig eth1 down

Manually building and deleting routing table entries

The *route* command is used to do both of these operations.

For example, to add a default gateway manually, do

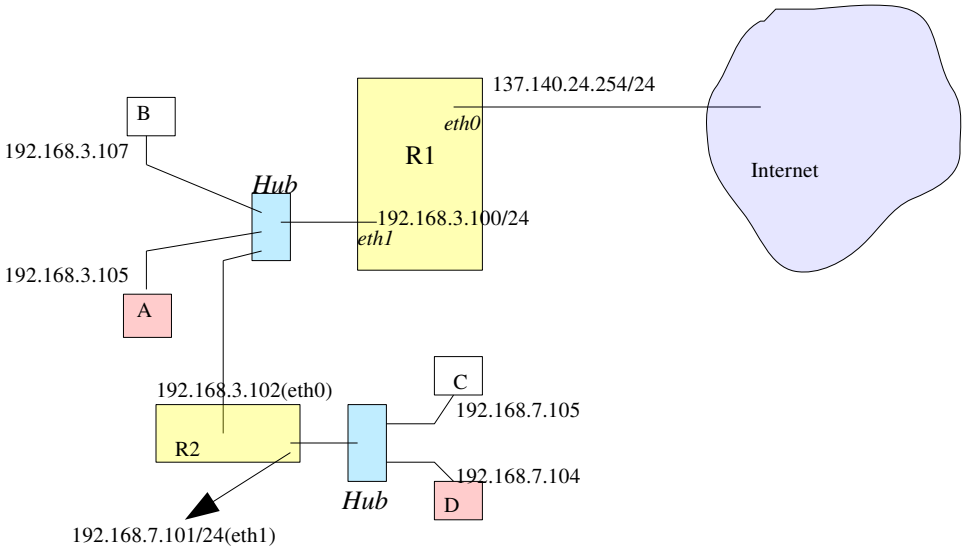
root> route add default gw 192.168.1.105 netmask 255.255.255.0

To delete the default route, do

root>route del default

The man pages for route and ifconfig commands give a number of command line options you can use with these commands.

Exercise: In the following network scenario, construct routing tables for R1, R2, A and D. Packets from every host should be able to reach 192.168.3.0/24 and 192.168.7.0/24 networks, as well as the outside Internet through a default gateway 137.140.24.250.



Additional material on Linux IP layer routing can be found at
http://linux-ip.net/html/index.html