Summary

Routing table can be viewed using `netstat -r` (with host resolution) and `netstat- rn` (without). See netstat

Route command can be used to:

1. **Add a route** `# route add net 128.50.3.0 tuna`
2. **Add a default route** `# route add default tomato 1`
3. Manual manipulation of the routing table: `[host|net] dest. [ gateway`
4. **Delete a route** `# route delete net 128.50.3.0 sword-r`
5. **Lookup and display the route for a destination** `# route get 128.50.2.0`
6. **Get routing reports continuously**

   ```
   # route monitor
   ```

7. **Flush the routing table** # route flush

8. **Add the multicast path for 224.0.0.0** `# route add 224.0.0.0 `uname -n` 0`
9. **Use the "route add net" command with the -netmask option to make the route command to take the netmask specified on the command line :**

   # route add net 192.168.68.0 128.50.1.250 1 -netmask 255.255.255.192

Introduction

Routing took place at the Internet layer of TCP/IP protocol.

1. TCP/IP Layers
    1. Application Layer
    2. Transport Layer
    3. **Internet Layer <- routing**
    4. Network Interface Layer
2. Hardware Layer.

A simple definition of routing is "learning how to get from here to there." (routes are both source and destination *dependent;* i.e., that knowing how to get there isn't enough, you have to know where you are starting from as well.) In some cases, the term *routing* is used in a very strict sense to refer *only* to the process of obtaining and distributing information ("learning"), but not to the process of using that information to actually get from one place to another (for which a different term, *forwarding*,is reserved). Since it is difficult to grasp the usefulness of information that is acquired but never used, we employ the term *routing* to refer in general to all the things that are done to discover and advertise paths from here to there and to actually move packets from here to there when necessary.

There are two types of routing: *direct* and *indirect*.

- **Direct Routing**. Direct routing occurs when the destination host is attached to the same physical network as the source host. The source host can send the IP datagram using the physical network frame without any involvement from the router.

- **Indirect Routing.** Indirect routing is used when the network numbers of the source and destination do not match. In this case the packet must be forwarded to a node that knows how to reach the destination (a router).
  The address of the first router (the first hop) is called an *indirect route.* A *default route* contains the router information to use for all destinations that do not have an explicit route table entry.

Kernel Routing Algorithm

When implementing routing Solaris kernel:

1. **Checks local LAN for destination hosts.** The kernel extracts the destination IP address from the IP datagram and computes the destination network number. The destination network number is then compared with the network numbers of all local interfaces (an interface physically attached to the system) for a match. If one of the destination network numbers matches that of a local interface network number, the kernel encapsulates the packet and sends it through the matching local interface for delivery.
2. **Checks routing table for matching IP host address.** If no local interface network number matches the destination network number, the kernel searches the routing table for a matching host IP address.
3. **Checks routing table for matching network number.** If no specific IP host address matches the destination IP address, the kernel searches the routing table for a matching network number. If found, the kernel sets the destination Ethernet address to that of the corresponding router. It completes the encapsulation of the packet, leaving the destination IP address unchanged, so that the next router will execute the routing algorithm again.
4. **Checks for a default entry in the routing table.** If there is no matching network number in the routing table, the kernel checks for a default entry in the routing table. If found, the kernel encapsulates the packet, setting the destination Ethernet address to that of the default router, leaving the destination IP address unchanged, and delivers the packet through the interface that is local to the default router.
5. **If there is route to host, generate ICMP error message.** If no matching address is found and no default router entry is found in the routing table, the kernel cannot forward the packet and an error message from **ICMP** is generated. The error message will state No route to host or network is unreachable.

   The ICMP handles control and error messages. ICMP on a router or gateway sends reports of problems to the original source. ICMP also includes an echo request or reply that is used to test whether a destination is reachable or not. The ping command uses this protocol.

   ICMP redirects are most commonly used when a host is using default routing. If the router determines a more efficient way to forward the packet, it redirects the datagram using the best route and reports the correct route to the sender.

   The sending host's route table is updated with the new information. The drawback to this method of routing is that for every ICMP redirect, there is a separate entry in the sending host's route table. This can lead to a large route table. However, it also ensures that the packets going to all reachable hosts are taking the shortest route.

   Common ICMP messages:

   - Echo request and reply messages from ping command
   - Report unreachable destinations

- Control congestion and datagram flow
- Route change requests from gateways to hosts
- Detect circular or excessively long routes
- Clock synchronization and transit time estimation
- Report other problems

Interior vs. Exterior Routing protocols

A single routing protocol cannot efficiently handle all situations because networks can be connected in many different ways.

An autonomous system (AS) is a collection of networks and routers under a single administrative control. This broad definition was incorporated into the Internet in an attempt to reduce excessively large route tables.

An autonomous system number is a unique 16-bit address that is assigned by the Internet Corporation for Assigned Names and Numbers (ICANN).

An autonomous system's exterior routers maintain route tables by using autonomous system numbers that represent exterior routes because the numbers create unique paths. An autonomous system's interior routers also have interior route entries in their route tables for subnets within the organization.

There are two major types of routing protocols:

- **interior routing protocols (IGP)**: are used between organizations or sites, for example, a large wide area network (WAN), such as the Internet or a large corporation's intranet. IGP is a route table protocol within an autonomous system. IGPs typiclly are used within an organization or an organization's site.
- **exterior routing protocols** that are used between autonomous systems. EGP and the Border Gateway Protocol (BGP) are the two principal protocols that exchange route table information among autonomous systems.

Interior Routing Protocols

Interior routing protocols pass route table information within an autonomous system (AS).T here are to major interior routing protocols:

- **RIP** RIP is a ***distance-vector*** protocol that exchanges route information between IP routers. Distance-vector algorithms obtain their name from the fact that they compute the least-cost path by using information that is exchanged with other routers that describes reachable networks with their distances in the form of hop counts.
- **Open Shortest Path First (OSPF) Protocol.** OSPF is a ***link-state*** protocol. OSPF maintains a map of the network topology instead of computing route paths that are based on distance vectors in the way that RIP computes the route paths. OSPF provides a global view of the network and provides the shortest path choices on routes. The map on each OSPF router is updated regularly.

Exterior Routing Protocols

An exterior routing protocol is a routing protocol that communicates routes between autonomous systems. EGP and the Border Gateway Protocol (BGP) are the two principal protocols that exchange route table information among autonomous systems.

- **EGP** was developed in the early 1980s. It was a ***distance vector*** protocol. The concept of an autonomous system developed out of the research and development of EGP.

- **BGP** was developed in the mid 1990s to replace EGP. ***BGP replaces the distance-vector algorithm of EGP with a path-vector algorithm.*** The path vector that is implemented by BGP causes the route table information to include a complete path (all autonomous system numbers) from the source to the destination. This eliminates the possibility of looping problems that might arise from complex network topologies, such as the Internet. A loop is detected by BGP when the path it receives has an autonomous system listed twice. If this occurs, BGP generates an error condition.

Classless and Classful Routing Protocols

For routers in a variably subnetted network to properly update each other, they must send masks in their routing updates. Without subnet information in the routing updates, routers would have nothing but the address class and their own subnet mask to go on. Only routing protocols that ignore the rules of address class and use classless prefixes work properly with VLSM. Table below lists common classful and classless routing protocols.

Classful Routing Protocols	Classless Routing Protocols
RIP Version 1 (distance vector)	RIP Version 2 (distance vector)
IGRP	EIGRP
EGP (distance vector)	OSPF (link state)
BGP3 (path vector)	IS-IS (link state)
	BGP4 (path vector)

Note: Only Solaris 10 supports RIP version 2 out of the box.

Routing Table

Routes are kept is a special table called *routing table*. The Solaris kernel stores it in memory. It contains two types of entries:

- **Static routes** are permnent entries in the route table. After such a route is in the table, you can only remove it manually. The most common static entries are the direct routes that a system creates to its local networks. During boot, the ifconfig utility updates the route table with static entries for the networks that are directly connected to the local network interfaces. Therefore, even in the single-user mode, a system can route directly to the local networks because the interfaces are initialized by the ifconfig utility. Static routes can also be added to your system's route table manually by the /etc/defaultrouter file or by entries placed in the /etc/gateways file. These static entries define the network destinations that are not directly connected but are reachable through another system or device called a router.
- **Dynamic routes** are added to or removed from the route table by various processes, such as the **in.routed** or **in.rdisc** processes. When the route table is updated with information about routers and other reachable networks, the identified router can forward or deliver datagrams to these networks. In Solaris 9 the **/etc/rc2.d/S69inet** script starts the two daemons that implement dynamic routing at run level 2:
    - **The Routing Information Protocol (RIP)** is implemented by the **in.routed** process (in Solaris 10 RIP2 is used). It also implements ICMP Router discovery protocol.
    - **The Router Discovery Protocol (RDISC)** is implemented by the **in.rdisc** process.

  Routers advertise the networks that they know about. Other hosts and routers listen to these periodic announcements and update their route table with the most current and correct information. Only those entries calculated to be the best paths to a network destination remain in the route table.

Displaying the Route Table

To display the contents of a system's route table without interpreting the names of the systems, use the netstat utility with the -r and -n options:

- The -r option causes the route table to be displayed.
- The -n option causes the IP addresses to be displayed instead of resolving them to names.

```
# netstat -rn

Routing Table: IPv4
  Destination           Gateway           Flags  Ref     Use
Interface
-------------------- -------------------- ----- ----- ---------- --------
-
default              10.201.12.1          UG        1     348856
10.201.12.0          10.201.13.251        U         1      20398 bge0
10.201.28.0          10.201.29.251        U         1       3170 bge2
224.0.0.0            10.201.13.251        U         1          0 bge0
127.0.0.1            127.0.0.1            UH       19     319784 lo0
```

Introducing Route Table Entries

Routing table entries:

- **Destination** The destination network or host address.
- **Gateway** The system that delivers or forwards the datagram.
- **Flags** The status of this route. This field uses the following flags:
  - U–The interface is up.
  - H–The destination is a system, not a network.
  - G–The delivery system is another system (an indirect path).
  - D–The entry was added dynamically by an ICMP redirect.
- **Ref** The current number of routes that share the same network interface (Ethernet) address.
- **Use** The number of datagrams that are using this route. For the localhost entry, it is a snapshot of the number of datagrams that are received.
- **Interface** The local interface that reaches the destination.

Static Routes

You can configure a route that does not change or time-out. This type of route is called a static route.

You can use the route utility to define a static direct route. A static route is a route that is not automatically removed by the **in.routed** process if a more efficient route is identified. The **ifconfig** utility ***initially builds the direct route entries when the network interface is configured during system startup.*** To view the results of the utility, perform the command: **netstat -r**

The **localhost** entry in the local routing table is a loopback route to the local host that is created when the lo0 pseudo interface is configured.

Configuring Static Routes Using route Utility

The Solaris route command enables manual manipulation of the route table The syntax of the command is somewhat complicated:

route [-fn] add|delete [net|host|default] destination gateway

With keywords **add** and **delete** the default for optional **[net|host|default]** troika is **host**.

For example, to add a direct static route between the systems **alpha** and **beta**, perform a command similar to the following:

```
# route add beta alpha
add host beta: gateway alpha
```

This is equivalent to:

```
# route add host beta alpha
```

To delete the route between **beta** and **alpha,** perform a command similar to the following:

```
# route delete beta alpha
delete host beta: gateway alpha
```

To define a default route using the system **sigma** , perform a command similar to the following:

```
# route add default sigma
add net default: gateway sigma
```

The **route** utility can also to retrieve information about a specific route. For example, to retrieve information about the default route:

```
# route get default
   route to: default
destination: default
       mask: default
    gateway: 10.201.12.1
  interface: bge0
      flags:
 recvpipe  sendpipe  ssthresh    rtt,ms rttvar,ms  hopcount      mtu     expire
       0         0         0         0         0         0      1500         0
```

To change the route table, use the change option with the route utility. For example, to change the default route from sigma to gamma, perform a command similar to the following:

```
# route change default gamma
change net default: gateway gamma
```

To continuously report any changes to the route table, route lookup misses, or suspected network partitionings, use the **monitor** option:

```
# route monitor
got message of size 124
RTM_DELETE: Delete Route: len 124, pid: 633, seq 1, errno 0,
flags:<UP,GATEWAY,DONE,STATIC>
locks: inits:
sockaddrs: <DST,GATEWAY,NETMASK>


192.168.3.0 alphaext 255.255.255.0
```

To flush (remove) the route table ***of all gateway entries***, use the **flush** option ***(only indirect routing table entries would be removed***):

```
# route flush

192.168.9 beta done
two beta done
two alphaext done
default 172.20.4.248 done
```

you can also flush the route table before any command with **-f** option:

```
# route -f add net 192.168.4.0 alphaext
add net 192.168.4.0: gateway alphaext
```

In case you accidentally deleted mulcast entry you can restore it manually (you can consult the command syntax in the **/etc/rc2.d/S72inetsvc** startup file). For example to add a route to the multicast address range of 224 through 239, perform the command:

```
# route add 224.0/4 ‘uname -n‘
```

To define a route that uses a specific netmask, use the netmask option with the route utility. For example, to add a route to the 192.168.3.0 network that uses a netmask of 255.255.255.224, perform the command:

```
# route add net 192.168.3.0 deltaext -netmask 255.255.255.224
add net 192.168.3.0: gateway deltaext
```

More concise way of doing the same would be to specify 192.168.3.0/27 **:**

```
# route add net 192.168.3.0/27 deltaext
add net 192.168.3.0/27: gateway deltaext
```

Note: The **in.routed** process does not automatically detect route table changes. Therefore, do not perform changes while the **in.routed** process is running. Instead, shut down the **in.routed** process, make the required changes, and then restart the **in.routed** process. This ensures that the **in.routed** process use the latest version of the routing table.

You can use the **route** utility to define a static routes. A static route is a route that is not automatically removed by the **in.routed** process if a more efficient route is identified.

Associating Network Name and Network Number

In Solaris the file **/etc/inet/networks** associates a network name to a network number. The file **/etc/networks** is a symbolic link to the **/etc/inet/networks**

There are three fields in the **/etc/inet/networks** file:

- network name
- network number
- nicknames.

```
# tail -2 /etc/inet/networks
```

```
one 192.168.1 one
two 192.168.2 two
```

To add a route to the network that is not defined in the **/etc/inet/networks** file you can use vi or command **route**:

```
# route add net 192.168.3.0 192.168.30.31
add net 192.168.3.0: gateway 192.168.30.31
```

To add a route to the network defined as "two" network you can use the command:

```
# route add net two 192.168.30.31
add net two: gateway 192.168.30.31
```

To view how defined networks are displayed in the output from the netstat utility, use the netstat utility with the -r option:

```
# netstat -r
```

Observe how the destination networks are now displayed by name instead of by network address. Loopback address is replaced by its entry from the **/etc/inet/hosts**

Here is Solaris 9 man page for this file:

networks - network name database

SYNOPSIS

/etc/inet/networks /etc/networks

DESCRIPTION

The networks file is a local source of information regarding the networks which comprise the Internet. The networks file can be used in conjunction with, or instead of, other net- works sources, including the NIS maps networks.byname and networks.byaddr and the NIS+ table networks. Programs use the getnetbyname(3SOCKET) routines to access this informa- tion. The network file has a single line for each network, with the following information: *official-network-name network-number aliases* Items are separated by any number of SPACE or TAB charac- ters. A `#' indicates the beginning of a comment. Characters up to the end of the line are not interpreted by routines which search the file. This file is normally created from the official network database maintained at the Network Information Control Center (NIC), though local changes may be required to bring it up to date regarding unofficial aliases and/or unknown networks. Network numbers may be specified in the conventional dot (`.') notation using the inet_network routine from the Internet address manipulation library, inet(7P). Network names may contain any printable character other than a field delimiter, NEWLINE, or comment character.

SEE ALSO

getnetbyaddr(3SOCKET), getnetbyname(3SOCKET), inet(3SOCKET), nsswitch.conf(4), inet(7P)

NOTES

The official SVR4 name of the networks file is /etc/inet/networks. The symbolic link /etc/networks exists for BSD compatibility. The network number in networks database is the host address shifted to the right by the number of 0 bits in the address mask. For example, for the address 24.132.47.86 that has a mask of fffffe00, its network number is 803351. This is obtained when the address is shifted right by 9 bits. The address maps to 12.66.23. The trailing 0 bits should not be SunOS 5.9 Last change: 17 Jan 2002 1 File Formats networks(4) specified. The network number here is different from that described in netmasks(4). For this example, the entry in netmasks would be 24.132.46.0fffffe00.

Default routes

A default route is a route table entry that allows a host to define default routers to use if no other specific route is available. The default routers must be reliable. There is no need to define every reachable network. All indirectly connected packet destinations go to the default router.

A default router can be identified by creating the **/etc/defaultrouter** file that contains hostname or IP address entries that identify one or more routers. Upon rebooting, this prevents the startup of the **in.routed** and **in.rdisc** daemons. Default route table entries may also be added by the **in.rdisc** daemon.

Some advantages of default routing are:

- The **/etc/defaultrouter** file prevents unneeded routing processes from starting.
- The default entries result in a smaller route table, which reduces the processing time spent on each IP datagram.
- Multiple default routers can be identified, which eliminate single points-of-failure within a network.
- Systems that use default route entries do not depend on actual routing protocols.

Some disadvantages of default routing are:

- The default entries created by the **/etc/defaultrouter** file or the **route** utility are always present, even when the default router is not available. The system does not learn about other possible routes.
- All systems must have a local **/etc/defaultrouter** file properly configured because this file cannot be administered by a name service. This can be an administrative problem on large, evolving networks.

A default route is a route table entry that defines the default routers to use if no other specific route is available. Default route entries can be either static entries or dynamic entries. The default routers must be reliable. You do not need to define every reachable network because datagrams that are addressed to non-local destinations use a default router in the absence of an explicit route.

Using the /etc/gateways File to Add Static Routes

During the initialization the **in.routed** router process reads the **/etc/gateways** file if it exists. ***This file can contains additional static routes and represents an alternative way to add such routes.*** The fields in the **/etc/gateways** file are:

net | host destination gateway gateway metric hops passive | active

For example:

```
# cat /etc/gateways
net 192.168.4.0 gateway gammaext metric 2 passive
```

Note – if the host name is used it should be resolved in /etc/hosts

Directives in the gateways file can also be used to prevent in.routed daemon from sending/recievieng advertizements from the specified interface. Use the **noripin** directive when you want your system to ignore route information that can be received on a specific interface. For example, to ignore route information received on the **qfe3** interface, use the following **noripin** directive in the gateways file:

```
noripin qfe3
```

You can use the **noripout** directive to prevent a multihomed system (system with multiple physical interfaces) from acting as a router and advertising routes. For example, to ensure that no route information is sent out of the qfe3 interface, use the following **noripout** directive in the gateways file:

```
noripout qfe3
```

You can choose to use both the **noripin** and **noripout** directives or replace them with a single **norip** directive. For example, to ignore route information and to not allow route information to be sent out of the qfe3 interface, use the following **norip** directive in the gateways file:

```
norip qfe3
```

Selected routing algorithms

Routing algorithms should:

- Scale well
- Support many different subnetwork types and multiple qualities of service
- Adapt to topology changes quickly and efficiently (i.e., with minimum overhead and complexity)
- Provide controls that facilitate the "safe" connection of multiple organizations

As we already briefly discussed above classic routing algorithms are

- **RIP**
- **RDISC**
- OSPF
- **BGP Protocol**