NAME

routed, **rdisc** - network RIP and router discovery routing daemon

SYNOPSIS

routed [**-isqdghmpAtv**] [**-T** *tracefile*] [**-F** *net*[/mask[,metric]]] [**-P** *parms*]

DESCRIPTION

The **routed** utility is a daemon invoked at boot time to manage the network routing tables. It uses Routing Information Protocol, RIPv1 (RFC 1058), RIPv2 (RFC 1723), and Internet Router Discovery Protocol (RFC 1256) to maintain the kernel routing table. The RIPv1 protocol is based on the reference 4.3BSD daemon.

It listens on the udp(4) socket for the route(8) service (see services(5)) for Routing Information Protocol packets. It also sends and receives multicast Router Discovery ICMP messages. If the host is a router, **routed** periodically supplies copies of its routing tables to any directly connected hosts and networks. It also advertises or solicits default routes using Router Discovery ICMP messages.

When started (or when a network interface is later turned on), **routed** uses an AF_ROUTE address family facility to find those directly connected interfaces configured into the system and marked "up". It adds necessary routes for the interfaces to the kernel routing table. Soon after being first started, and provided there is at least one interface on which RIP has not been disabled, **routed** deletes all pre-existing non-static routes in kernel table. Static routes in the kernel table are preserved and included in RIP responses if they have a valid RIP -hopcount (see route(8)).

If more than one interface is present (not counting the loopback interface), it is assumed that the host should forward packets among the connected networks. After transmitting a RIP *request* and Router Discovery Advertisements or Solicitations on a new interface, the daemon enters a loop, listening for RIP request and response and Router Discovery packets from other hosts.

When a *request* packet is received, **routed** formulates a reply based on the information maintained in its internal tables. The *response* packet generated contains a list of known routes, each marked with a "hop count" metric (a count of 16 or greater is considered "infinite"). The advertised metric for a route reflects the metrics associated with interfaces (see ifconfig(8)) though which it is received and sent, so setting the metric on an interface is an effective way to steer traffic. See also **adj_inmetric** and **adj_outmetric** parameters below.

Responses do not include routes with a first hop on the requesting network to implement in part *split-horizon*. Requests from query programs such as rtquery(8) are answered with the complete table.

The routing table maintained by the daemon includes space for several gateways for each destination to

speed recovery from a failing router. RIP *response* packets received are used to update the routing tables provided they are from one of the several currently recognized gateways or advertise a better metric than at least one of the existing gateways.

When an update is applied, **routed** records the change in its own tables and updates the kernel routing table if the best route to the destination changes. The change in the kernel routing table is reflected in the next batch of *response* packets sent. If the next response is not scheduled for a while, a *flash update* response containing only recently changed routes is sent.

In addition to processing incoming packets, **routed** also periodically checks the routing table entries. If an entry has not been updated for 3 minutes, the entry's metric is set to infinity and marked for deletion. Deletions are delayed until the route has been advertised with an infinite metric to ensure the invalidation is propagated throughout the local internet. This is a form of *poison reverse*.

Routes in the kernel table that are added or changed as a result of ICMP Redirect messages are deleted after a while to minimize *black-holes*. When a TCP connection suffers a timeout, the kernel tells **routed**, which deletes all redirected routes through the gateway involved, advances the age of all RIP routes through the gateway to allow an alternate to be chosen, and advances of the age of any relevant Router Discovery Protocol default routes.

Hosts acting as internetwork routers gratuitously supply their routing tables every 30 seconds to all directly connected hosts and networks. These RIP responses are sent to the broadcast address on nets that support broadcasting, to the destination address on point-to-point links, and to the router's own address on other networks. If RIPv2 is enabled, multicast packets are sent on interfaces that support multicasting.

If no response is received on a remote interface, if there are errors while sending responses, or if there are more errors than input or output (see netstat(1)), then the cable or some other part of the interface is assumed to be disconnected or broken, and routes are adjusted appropriately.

The *Internet Router Discovery Protocol* is handled similarly. When the daemon is supplying RIP routes, it also listens for Router Discovery Solicitations and sends Advertisements. When it is quiet and listening to other RIP routers, it sends Solicitations and listens for Advertisements. If it receives a good Advertisement and it is not multi-homed, it stops listening for broadcast or multicast RIP responses. It tracks several advertising routers to speed recovery when the currently chosen router dies. If all discovered routers disappear, the daemon resumes listening to RIP responses. It continues listening to RIP while using Router Discovery if multi-homed to ensure all interfaces are used.

The Router Discovery standard requires that advertisements have a default "lifetime" of 30 minutes. That means should something happen, a client can be without a good route for 30 minutes. It is a good

idea to reduce the default to 45 seconds using **-P rdisc_interval=45** on the command line or **rdisc_interval=45** in the */etc/gateways* file.

While using Router Discovery (which happens by default when the system has a single network interface and a Router Discover Advertisement is received), there is a single default route and a variable number of redirected host routes in the kernel table. On a host with more than one network interface, this default route will be via only one of the interfaces. Thus, multi-homed hosts running with **-q** might need **no_rdisc** described below.

See the **pm_rdisc** facility described below to support "legacy" systems that can handle neither RIPv2 nor Router Discovery.

By default, neither Router Discovery advertisements nor solicitations are sent over point to point links (e.g. PPP). The netmask associated with point-to-point links (such as SLIP or PPP, with the IFF_POINTOPOINT flag) is used by **routed** to infer the netmask used by the remote system when RIPv1 is used.

The following options are available:

-i allow **routed** to accept a RIP request from non-router node. When specified once, **routed** replies to a route information query from neighbor nodes. When specified twice, it replies to a query from remote nodes in addition. rtquery(8) utility can be used to send a request.

This feature is disabled by default because of a risk of reflection attack though it is useful for debugging purpose.

-s force **routed** to supply routing information. This is the default if multiple network interfaces are present on which RIP or Router Discovery have not been disabled, and if the kernel switch ipforwarding=1.

-q is the opposite of the **-s** option. This is the default when only one interface is present. With this explicit option, the daemon is always in "quiet-mode" for RIP and does not supply routing information to other computers.

-d do not run in the background. This option is meant for interactive use.

-g used on internetwork routers to offer a route to the "default" destination. It is equivalent to **-F 0/0,1** and is present mostly for historical reasons. A better choice is **-P pm_rdisc** on the command line or **pm_rdisc** in the */etc/gateways* file, since a larger metric will be used, reducing the spread of the potentially dangerous default route. This is typically used on a gateway to the

Internet, or on a gateway that uses another routing protocol whose routes are not reported to other local routers. Notice that because a metric of 1 is used, this feature is dangerous. It is more commonly accidentally used to create chaos with a routing loop than to solve problems.

-h cause host or point-to-point routes to not be advertised, provided there is a network route going the same direction. That is a limited kind of aggregation. This option is useful on gateways to Ethernets that have other gateway machines connected with point-to-point links such as SLIP.

-m cause the machine to advertise a host or point-to-point route to its primary interface. It is useful on multi-homed machines such as NFS servers. This option should not be used except when the cost of the host routes it generates is justified by the popularity of the server. It is effective only when the machine is supplying routing information, because there is more than one interface. The **-m** option overrides the **-q** option to the limited extent of advertising the host route.

-A do not ignore RIPv2 authentication if we do not care about RIPv2 authentication. This option is required for conformance with RFC 1723. However, it makes no sense and breaks using RIP as a discovery protocol to ignore all RIPv2 packets that carry authentication when this machine does not care about authentication.

-t increase the debugging level, which causes more information to be logged on the tracefile specified with **-T** or standard out. The debugging level can be increased or decreased with the *SIGUSR1* or *SIGUSR2* signals or with the rtquery(8) command.

-T *tracefile*
increases the debugging level to at least 1 and causes debugging information to be appended to the trace file. Note that because of security concerns, it is wisest to not run **routed** routinely with tracing directed to a file.

-v display and logs the version of daemon.

-F *net[/mask][,metric]*
minimize routes in transmissions via interfaces with addresses that match *net/mask*, and synthesizes a default route to this machine with the *metric*. The intent is to reduce RIP traffic on slow, point-to-point links such as PPP links by replacing many large UDP packets of RIP information with a single, small packet containing a "fake" default route. If *metric* is absent, a value of 14 is assumed to limit the spread of the "fake" default route. This is a dangerous feature that when used carelessly can cause routing loops. Notice also that more than one interface can match the specified network number and mask. See also **-g**.

-P *parms*

is equivalent to adding the parameter line *parms* to the */etc/gateways* file.

Any other argument supplied is interpreted as the name of a file in which the actions of **routed** should be logged. It is better to use **-T** instead of appending the name of the trace file to the command.

The **routed** utility also supports the notion of "distant" *passive* or *active* gateways. When **routed** is started, it reads the file */etc/gateways* to find such distant gateways which may not be located using only information from a routing socket, to discover if some of the local gateways are *passive*, and to obtain other parameters. Gateways specified in this manner should be marked passive if they are not expected to exchange routing information, while gateways marked active should be willing to exchange RIP packets. Routes through *passive* gateways are installed in the kernel's routing tables once upon startup and are not included in transmitted RIP responses.

Distant active gateways are treated like network interfaces. RIP responses are sent to the distant *active* gateway. If no responses are received, the associated route is deleted from the kernel table and RIP responses advertised via other interfaces. If the distant gateway resumes sending RIP responses, the associated route is restored.

Such gateways can be useful on media that do not support broadcasts or multicasts but otherwise act like classic shared media like Ethernets such as some ATM networks. One can list all RIP routers reachable on the HIPPI or ATM network in */etc/gateways* with a series of "host" lines. Note that it is usually desirable to use RIPv2 in such situations to avoid generating lists of inferred host routes.

Gateways marked *external* are also passive, but are not placed in the kernel routing table nor are they included in routing updates. The function of external entries is to indicate that another routing process will install such a route if necessary, and that other routes to that destination should not be installed by **routed**. Such entries are only required when both routers may learn of routes to the same destination.

The */etc/gateways* file is comprised of a series of lines, each in one of the following two formats or consist of parameters described later. Blank lines and lines starting with '#' are comments.

net *Nname[/mask]* **gateway** *Gname* **metric** *value* <**passive** | **active** | **extern**>

host *Hname* **gateway** *Gname* **metric** *value* <**passive** | **active** | **extern**>

Nname or *Hname* is the name of the destination network or host. It may be a symbolic network name or an Internet address specified in "dot" notation (see inet(3)). (If it is a name, then it must either be defined in */etc/networks* or */etc/hosts*, or a method in nsswitch.conf(5) must be able to resolve it.)

Mask is an optional number between 1 and 32 indicating the netmask associated with *Nname*.

Gname is the name or address of the gateway to which RIP responses should be forwarded.

Value is the hop count to the destination host or network.

Host *hname* is equivalent to **net** *nname/32*.

One of the keywords **passive**, **active** or **external** must be present to indicate whether the gateway should be treated as **passive** or **active** (as described above), or whether the gateway is **external** to the scope of the RIP protocol.

As can be seen when debugging is turned on with **-t**, such lines create pseudo-interfaces. To set parameters for remote or external interfaces, a line starting with **if=alias(Hname)**, **if=remote(Hname)**, etc. should be used.

Parameters

Lines that start with neither "net" nor "host" must consist of one or more of the following parameter settings, separated by commas or blanks:

if=*ifname*
: indicates that the other parameters on the line apply to the interface name *ifname*.

subnet=*nname*[/*mask*][,*metric*]
: advertises a route to network *nname* with mask *mask* and the supplied metric (default 1). This is useful for filling "holes" in CIDR allocations. This parameter must appear by itself on a line. The network number must specify a full, 32-bit value, as in 192.0.2.0 instead of 192.0.2.

  Do not use this feature unless necessary. It is dangerous.

ripv1_mask=*nname*/*mask1*,*mask2*
: specifies that netmask of the network of which *nname*/*mask1* is a subnet should be *mask2*. For example, "ripv1_mask=192.0.2.16/28,27" marks 192.0.2.16/28 as a subnet of 192.0.2.0/27 instead of 192.0.2.0/24. It is better to turn on RIPv2 instead of using this facility, for example with **ripv2_out**.

passwd=*XXX[|KeyID[start|stop]]*
: specifies a RIPv2 cleartext password that will be included on all RIPv2 responses sent, and checked on all RIPv2 responses received. Any blanks, tab characters, commas, or '#', '|', or NULL characters in the password must be escaped with a backslash (\). The common escape sequences \n, \r, \t, \b, and \xxx have their usual meanings. The **KeyID** must be unique but is ignored for cleartext passwords. If present, **start** and **stop** are timestamps in the form

year/month/day@hour:minute. They specify when the password is valid. The valid password with the most future is used on output packets, unless all passwords have expired, in which case the password that expired most recently is used, or unless no passwords are valid yet, in which case no password is output. Incoming packets can carry any password that is valid, will be valid within the next 24 hours, or that was valid within the preceding 24 hours. To protect the secrets, the passwd settings are valid only in the */etc/gateways* file and only when that file is readable only by UID 0.

md5_passwd=*XXX|KeyID[start|stop]*
specifies a RIPv2 MD5 password. Except that a **KeyID** is required, this keyword is similar to **passwd**.

no_ag turns off aggregation of subnets in RIPv1 and RIPv2 responses.

no_super_ag
turns off aggregation of networks into supernets in RIPv2 responses.

passive
marks the interface to not be advertised in updates sent via other interfaces, and turns off all RIP and router discovery through the interface.

no_rip
disables all RIP processing on the specified interface. If no interfaces are allowed to process RIP packets, **routed** acts purely as a router discovery daemon.

Note that turning off RIP without explicitly turning on router discovery advertisements with **rdisc_adv** or **-s** causes **routed** to act as a client router discovery daemon, not advertising.

no_rip_mcast
causes RIPv2 packets to be broadcast instead of multicast.

no_rip_out
causes no RIP updates to be sent.

no_ripv1_in
causes RIPv1 received responses to be ignored.

no_ripv2_in
causes RIPv2 received responses to be ignored.

ripv2_out
turns on RIPv2 output and causes RIPv2 advertisements to be multicast when possible.

ripv2 is equivalent to **no_ripv1_in** and **no_ripv1_out**. This enables RIPv2.

no_rdisc
disables the Internet Router Discovery Protocol.

no_solicit
disables the transmission of Router Discovery Solicitations.

send_solicit
specifies that Router Discovery solicitations should be sent, even on point-to-point links, which by default only listen to Router Discovery messages.

no_rdisc_adv
disables the transmission of Router Discovery Advertisements.

rdisc_adv
specifies that Router Discovery Advertisements should be sent, even on point-to-point links, which by default only listen to Router Discovery messages.

bcast_rdisc
specifies that Router Discovery packets should be broadcast instead of multicast.

rdisc_pref=*N*
sets the preference in Router Discovery Advertisements to the optionally signed integer *N*. The default preference is 0. Default routes with smaller or more negative preferences are preferred by clients.

rdisc_interval=*N*
sets the nominal interval with which Router Discovery Advertisements are transmitted to N seconds and their lifetime to 3*N.

fake_default=*metric*
has an identical effect to **-F** *net[/mask][=metric]* with the network and mask coming from the specified interface.

pm_rdisc
is similar to **fake_default**. When RIPv2 routes are multicast, so that RIPv1 listeners cannot

receive them, this feature causes a RIPv1 default route to be broadcast to RIPv1 listeners. Unless modified with **fake_default**, the default route is broadcast with a metric of 14. That serves as a "poor man's router discovery" protocol.

adj_inmetric=*delta*
adjusts the hop count or metric of received RIP routes by *delta*. The metric of every received RIP route is increased by the sum of two values associated with the interface. One is the adj_inmetric value and the other is the interface metric set with ifconfig(8).

adj_outmetric=*delta*
adjusts the hop count or metric of advertised RIP routes by *delta*. The metric of every received RIP route is increased by the metric associated with the interface by which it was received, or by 1 if the interface does not have a non-zero metric. The metric of the received route is then increased by the adj_outmetric associated with the interface. Every advertised route is increased by a total of four values, the metric set for the interface by which it was received with ifconfig(8), the **adj_inmetric** *delta* of the receiving interface, the metric set for the interface by which it is transmitted with ifconfig(8), and the **adj_outmetric** *delta* of the transmitting interface.

trust_gateway=*rname[|net1/mask1|net2/mask2|...]*
causes RIP packets from router *rname* and other routers named in other **trust_gateway** keywords to be accepted, and packets from other routers to be ignored. If networks are specified, then routes to other networks will be ignored from that router.

redirect_ok
allows the kernel to listen ICMP Redirect messages when the system is acting as a router and forwarding packets. Otherwise, ICMP Redirect messages are overridden and deleted when the system is acting as a router.

FILES

/etc/gateways for distant gateways

SEE ALSO

icmp(4), udp(4), rtquery(8)

Internet Transport Protocols, XSIS 028112, Xerox System Integration Standard.

HISTORY

The **routed** utility appeared in 4.2BSD.

BUGS

It does not always detect unidirectional failures in network interfaces, for example, when the output side fails.