NAME

route − show / manipulate the IP routing table

SYNOPSIS

route [**−CFvnNee**] [**−A** family |**−4**|**−6**]

route   [**−v**] [**−A** family |**−4**|**−6**] **add** [**−net**|**−host**] *target* [**netmask** *Nm*] [**gw** *Gw*] [**metric** *N*] [**mss** *M*] [**window** *W*] [**irtt** *I*] [**reject**] [**mod**] [**dyn**] [**reinstate**] [[**dev**] *If*]

route   [**−v**] [**−A** *family* |**−4**|**−6**] **del** [**−net**|**−host**] *target* [**gw** *Gw*] [**netmask** *Nm*] [**metric** *M*] [[**dev**] *If*]

route   [**−V**] [**−−version**] [**−h**] [**−−help**]

DESCRIPTION

Route manipulates the kernel's IP routing tables. Its primary use is to set up static routes to specific hosts or networks via an interface after it has been configured with the **ifconfig**(8) program.

When the **add** or **del** options are used, **route** modifies the routing tables. Without these options, **route** displays the current contents of the routing tables.

OPTIONS

−A *family*

use the specified address family (eg 'inet'). Use **route −−help** for a full list. You can use **−6** as an alias for **−−inet6** and **−4** as an alias for **−A inet**

−F   operate on the kernel's FIB (Forwarding Information Base) routing table. This is the default.

−C   operate on the kernel's routing cache.

−v   select verbose operation.

−n   show numerical addresses instead of trying to determine symbolic host names. This is useful if you are trying to determine why the route to your nameserver has vanished.

−e   use **netstat**(8)−format for displaying the routing table. **−ee** will generate a very long line with all parameters from the routing table.

del   delete a route.

add   add a new route.

target   the destination network or host. You can provide an addresses or symbolic network or host name. Optionally you can use **/***prefixlen* notation instead of using the **netmask** option.

−net   the *target* is a network.

−host   the *target* is a host.

netmask *NM*

when adding a network route, the netmask to be used.

gw *GW* route packets via a gateway.
              NOTE: The specified gateway must be reachable first. This usually means that you have to set up a static route to the gateway beforehand. If you specify the address of one of your local interfaces, it will be used to decide about the interface to which the packets should be routed to. This is a BS-Dism compatibility hack.

metric *M*

set the metric field in the routing table (used by routing daemons) to *M*. If this option is not specified the metric for inet6 (IPv6) address family defaults to '1', for inet (IPv4) it defaults to '0'. You should always specify an explicit metric value to not rely on those defaults - they also differ from iproute2.

mss *M*   sets MTU (Maximum Transmission Unit) of the route to *M* bytes. Note that the current implementation of the route command does not allow the option to set the Maximum Segment Size (MSS).

window *W*
          set the TCP window size for connections over this route to *W* bytes. This is typically only used on AX.25 networks and with drivers unable to handle back to back frames.

irtt *I*   set the initial round trip time (irtt) for TCP connections over this route to *I* milliseconds (1-12000). This is typically only used on AX.25 networks. If omitted the RFC 1122 default of 300ms is used.

reject   install a blocking route, which will force a route lookup to fail. This is for example used to mask out networks before using the default route. This is NOT for firewalling.

mod, dyn, reinstate
          install a dynamic or modified route. These flags are for diagnostic purposes, and are generally only set by routing daemons.

dev *If*   force the route to be associated with the specified device, as the kernel will otherwise try to determine the device on its own (by checking already existing routes and device specifications, and where the route is added to). In most normal networks you won't need this.

          If **dev** *If* is the last option on the command line, the word **dev** may be omitted, as it's the default. Otherwise the order of the route modifiers (**metric netmask gw dev**) doesn't matter.

EXAMPLES
route add −net 127.0.0.0 netmask 255.0.0.0 metric 1024 dev lo
          adds the normal loopback entry, using netmask 255.0.0.0 and associated with the "lo" device (assuming this device was previously set up correctly with **ifconfig**(8)).

route add −net 192.56.76.0 netmask 255.255.255.0 metric 1024 dev eth0
          adds a route to the local network 192.56.76.x via "eth0". The word "dev" can be omitted here.

route del default
          deletes the current default route, which is labeled "default" or 0.0.0.0 in the destination field of the current routing table.

route del −net 192.56.76.0 netmask 255.255.255.0
          deletes the route. Since the Linux routing kernel uses classless addressing, you pretty much always have to specify the netmask that is same as as seen in 'route -n' listing.

route add default gw mango
          adds a default route (which will be used if no other route matches). All packets using this route will be gatewayed through the address of a node named "mango". The device which will actually be used for that route depends on how we can reach "mango" - "mango" must be on directly reachable route.

route add mango sl0
          Adds the route to the host named "mango" via the SLIP interface (assuming that "mango" is the SLIP host).

route add −net 192.57.66.0 netmask 255.255.255.0 gw mango
          This command adds the net "192.57.66.x" to be gatewayed through the former route to the SLIP interface.

route add −net 224.0.0.0 netmask 240.0.0.0 dev eth0
> This is an obscure one documented so people know how to do it. This sets all of the class D (multicast) IP routes to go via "eth0". This is the correct normal configuration line with a multicasting kernel.

route add −net 10.0.0.0 netmask 255.0.0.0 metric 1024 reject
> This installs a rejecting route for the private network "10.x.x.x."

route -6 add 2001:0002::/48 metric 1 dev eth0
> This adds a IPv6 route with the specified metric to be directly reachable via eth0.

OUTPUT

> The output of the kernel routing table is organized in the following columns

Destination
> The destination network or destination host.

Gateway
> The gateway address or '*' if none set.

Genmask
> The netmask for the destination net; '255.255.255.255' for a host destination and '0.0.0.0' for the **default** route.

Flags Possible flags include
> **U** (route is **up**)
> **H** (target is a **host**)
> **G** (use **gateway**)
> **R** (**reinstate** route for dynamic routing)
> **D** (**dynamically** installed by daemon or redirect)
> **M** (**modified** from routing daemon or redirect)
> **A** (installed by **addrconf**)
> **C** (**cache** entry)
> **!** (**reject** route)

Metric The 'distance' to the target (usually counted in hops).

Ref Number of references to this route. (Not used in the Linux kernel.)

Use Count of lookups for the route. Depending on the use of −F and −C this will be either route cache misses (−F) or hits (−C).

Iface Interface to which packets for this route will be sent.

MSS Default maximum segment size for TCP connections over this route.

Window
> Default window size for TCP connections over this route.

irtt Initial RTT (Round Trip Time). The kernel uses this to guess about the best TCP protocol parameters without waiting on (possibly slow) answers.

HH (cached only)
> The number of ARP entries and cached routes that refer to the hardware header cache for the cached route. This will be −1 if a hardware address is not needed for the interface of the cached route (e.g. lo).

Arp (cached only)
> Whether or not the hardware address for the cached route is up to date.

FILES

/proc/net/ipv6_route
/proc/net/route
/proc/net/rt_cache

SEE ALSO

ifconfig(8), *netstat*(8), *arp*(8), *rarp*(8), *ip*(8)

HISTORY

Route for Linux was originally written by Fred N. van Kempen, <waltje@uwalt.nl.mugnet.org> and then modified by Johannes Stille and Linus Torvalds for pl15. Alan Cox added the mss and window options for Linux 1.1.22. irtt support and merged with netstat from Bernd Eckenfels.

AUTHOR

Currently maintained by Phil Blundell <Philip.Blundell@pobox.com> and Bernd Eckenfels <net-tools@lina.inka.de>.