NAME

rtentry - structure of an entry in the kernel routing table

SYNOPSIS

#include <sys/types.h>
#include <sys/socket.h>
#include <net/route.h>

DESCRIPTION

The kernel provides a common mechanism by which all protocols can store and retrieve entries from a central table of routes. Parts of this mechanism are also used to interact with user-level processes by means of a socket in the route(4) pseudo-protocol family. The *<net/route.h>* header file defines the structures and manifest constants used in this facility.

The basic structure of a route is defined by *struct rtentry*, which includes the following fields:

struct radix_node rt_nodes[2];
: Glue used by the radix-tree routines. These members also include in their substructure the key (i.e., destination address) and mask used when the route was created. The **rt_key**(*rt*) and **rt_mask**(*rt*) macros can be used to extract this information (in the form of a *struct sockaddr **) given a *struct rtentry **.

*struct sockaddr *rt_gateway*;
: The "target" of the route, which can either represent a destination in its own right (some protocols will put a link-layer address here), or some intermediate stop on the way to that destination (if the RTF_GATEWAY flag is set).

int rt_flags;
: See below. If the RTF_UP flag is not present, the **rtfree**() function will delete the route from the radix tree when the last reference drops.

int rt_refcnt;
: Route entries are reference-counted; this field indicates the number of external (to the radix tree) references.

*struct ifnet *rt_ifp*;

*struct ifaddr *rt_ifa*;
: These two fields represent the "answer", as it were, to the question posed by a route lookup; that is, they name the interface and interface address to be used in sending a

packet to the destination or set of destinations which this route represents.

u_long rt_mtu;
See description of rmx_mtu below.

u_long rt_weight;
See description of rmx_weight below.

u_long rt_expire;
See description of rmx_expire below.

counter64_t rt_pksent;
See description of rmx_pksent below.

*struct rtentry *rt_gwroute*;
This member is a reference to a route whose destination is *rt_gateway*. It is only used for RTF_GATEWAY routes.

struct mtx rt_mtx;
Mutex to lock this routing entry.

The following flag bits are defined:

RTF_UP	The route is not deleted.
RTF_GATEWAY	The route points to an intermediate destination and not the ultimate recipient; the *rt_gateway* and *rt_gwroute* fields name that destination.
RTF_HOST	This is a host route.
RTF_REJECT	The destination is presently unreachable. This should result in an EHOSTUNREACH error from output routines.
RTF_DYNAMIC	This route was created dynamically by **rtredirect**().
RTF_MODIFIED	This route was modified by **rtredirect**().
RTF_DONE	Used only in the route(4) protocol, indicating that the request was executed.
RTF_XRESOLVE	When this route is returned as a result of a lookup, send a report on the route(4) interface requesting that an external process perform resolution for this route.
RTF_STATIC	Indicates that this route was manually added by means of the route(8) command.
RTF_BLACKHOLE	Requests that output sent via this route be discarded.
RTF_PROTO1	
RTF_PROTO2	
RTF_PROTO3	Protocol-specific.

RTF_PINNED	Indicates that this route is immutable to a routing protocol.
RTF_LOCAL	Indicates that the destination of this route is an address configured as belonging to this system.
RTF_BROADCAST	Indicates that the destination is a broadcast address.
RTF_MULTICAST	Indicates that the destination is a multicast address.

Several metrics are supplied in *struct rt_metrics* passed with routing control messages via route(4) API. Currently only *rmx_mtu*, *rmx_expire*, and *rmx_pksent* metrics are supplied. All others are ignored.

The following metrics are defined by *struct rt_metrics*:

u_long rmx_locks;
Flag bits indicating which metrics the kernel is not permitted to dynamically modify.

u_long rmx_mtu;
MTU for this path.

u_long rmx_hopcount;
Number of intermediate systems on the path to this destination.

u_long rmx_expire;
The time (a la time(3)) at which this route should expire, or zero if it should never expire. It is the responsibility of individual protocol suites to ensure that routes are actually deleted once they expire.

u_long rmx_recvpipe;
Nominally, the bandwidth-delay product for the path *from* the destination *to* this system. In practice, this value is used to set the size of the receive buffer (and thus the window in sliding-window protocols like TCP).

u_long rmx_sendpipe;
As before, but in the opposite direction.

u_long rmx_ssthresh;
The slow-start threshold used in TCP congestion-avoidance.

u_long rmx_rtt;
The round-trip time to this destination, in units of RMX_RTTUNIT per second.

u_long rmx_rttvar;
The average deviation of the round-trip time to this destination, in units of RMX_RTTUNIT per second.

u_long rmx_pksent;
A count of packets successfully sent via this route.

u_long rmx_filler[4];
Empty space available for protocol-specific information.

SEE ALSO

route(4), route(8),

HISTORY

The *rtentry* structure first appeared in 4.2BSD. The radix-tree representation of the routing table and the *rt_metrics* structure first appeared in 4.3BSD-Reno.

AUTHORS

This manual page was written by Garrett Wollman.

BUGS

There are a number of historical relics remaining in this interface. The *rt_gateway* and *rmx_filler* fields could be named better.