Network Working Group
Request for Comments: 1476

R. Ullmann
Process Software Corporation

June 1993

RAP: Internet Route Access Protocol

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. It does not specify an Internet standard. Discussion and suggestions for improvement are requested. Please refer to the current edition of the "IAB Official Protocol Standards" for the standardization state and status of this protocol. Distribution of this memo is unlimited.

Abstract

This RFC describes an open distance vector routing protocol for use at all levels of the internet, from isolated LANs to the major routers of an international commercial network provider.

Table of Contents

1. Introduction

RAP is a general protocol for distributing routing information at all levels of the Internet, from private LANs to the widest-flung international carrier networks. It does not distinguish between "interior" and "exterior" routing (except as restricted by specific policy), and therefore is not as restricted nor complex as those protocols that have strict level and area definitions in their models.

The protocol encourages the widest possible dissemination of topology information, aggregating it only when limits of thrust, bandwidth, or administrative policy require it. Thus RAP permits aggressive use of resources to optimize routes where desired, without the restrictions inherent in the simplifications of other models.

While RAP uses IPv7 [RFC1475] addressing internally, it is run over both IPv4 and IPv7 networks, and shares routing information between them. A IPv4 router will only be able to activate and propagate routes that are defined within the local Administrative Domain (AD), loading the version 4 subset of the address into the local IP forwarding database.

1.1 Link-State and Distance-Vector

Of the two major classes of routing algorithm, link-state and distance vector, only distance vector seems to scale from the local network (where RIP is existence-proof of its validity) to large scale inter-domain policy routing, where the number of links and policies exceeds the ability of each router to map the entire network.

In between, we have OSPF, an open link state (specifically, using shortest-path-first analysis of the graph, hence the acronym) protocol, with extensive development in intra-area routing.

Since distance vector has proven useful at both ends of the range, it seems reasonable to apply it to the entire range of scales, creating a protocol that works automatically on small groups of LANs, but can apply fairly arbitrary policy in the largest networks.

This helps model the real world, where networks are not clearly divided into hierarchical domains with identifiable "border" routers, but have many links across organizational structure and over back fences.

1.2 Terminology

The RAP protocol propagates routes in the opposite direction to the travel of datagrams using the routes. To avoid confusion explaining the routing protocol, several terms are distinguished:

source	where datagrams come from, the source of the datagrams
destination	where datagrams go to, the destination of the datagrams
origin	where routing information originates, the router initially inserting route information into the RAP domain
target	where routing information goes, the target uses the information to send datagrams

1.3 Philosophy

Protocols should become simpler as they evolve.

2. RAP Protocol

The RAP protocol operates on TCP port 38, with peers opening a symmetric TCP connection between the RAP ports on each system. Thus only one RAP connection exists between any pair of peers.

RAP is also used on UDP port 38, as a peer discovery method. Hosts (i.e., non-routing systems) may listen to RAP datagrams on this port to discover local gateways. This is in addition to, or in lieu of, an Internet Standard gateway discovery protocol, which does not exist at this writing.

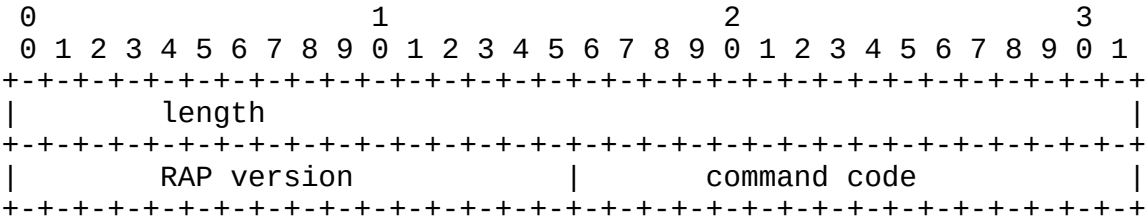
The peers then use RAP commands to send each other all routes available though the sending peer. This occurs as a full-duplex (i.e., simultaneous) exchange of information, with no acknowledgement of individual commands.

Once the initial exchange has been completed, the peers send only updates to routes, new routes, and purge commands to delete routes previously offered.

When the connection is broken, each system purges all routes that had been offered by the peer.

2.1 Command Header Format

Each RAP command starts with a header. The header contains a length field to identify the start of the next packet in the TCP stream, a version number, and the code for the command. On UDP, the length field does not appear: each UDP datagram must contain exactly one RAP command and not contain data or padding after the end of the command.



2.1.1 Length field

The length is a 32 bit unsigned number specifying the offset in bytes from the first byte of the length field of this command packet to the start of the length field of the next. The minimum value is 8. There is no specific limit to the length of a command packet; implementations MUST be able to at least count and skip over a packet

that is too large and then MAY send an error indication.

Each version of the protocol will profile what size should be considered the limit for senders, and what (larger) size should be considered by receivers to mean that the connection is insane: either unsynchronized or worse.

For version 1 of the protocol, senders MUST NOT send command packets greater than 16384 bytes. Receivers SHOULD consider packets that appear to be greater than 162144 bytes in length to be an indication of an unrecoverable error.

Note that these limits probably will not be approached in normal operation of version 1 of the protocol; receivers may reasonably decline to use routes described by 16K bytes of metrics and policy. But even the most memory-restricted implementation MUST be able to skip such a command packet.

2.1.2 RAP version

The version field is a 16 bit unsigned number. It identifies the version of RAP used for that command. Note that commands with different versions may be mixed on the same connection, although the usual procedure will be to do the serious protocol (exchanging route updates) only at the highest version common to both ends of the connection.

Each side starts the connection by sending a poll command, using the highest version supported and continues by using the highest version received in any command from the remote. The response to the poll will either be a no-operation packet at that version or an error packet at the highest version supported by the remote.

This document describes version 1 of the RAP protocol.

2.2 RAP Commands

There five simple RAP commands, described in the following sections.

2.2.1 No operation

The no operation command serves to reset the poll timer (see next section) of the receiver, or (as a side effect) to tell the receiver that a particular version is supported. It never contains option specific data and its length is always 8.

The no operation command is also used in a UDP broadcast to inform other systems that the sender is running RAP actively on the network

and is both a possible gateway and a candidate peer. If this command is being sent in response to a broadcast poll, it should be sent only to the poller.

A RAP process may send such broadcasts in a startup sequence, or it may persist indefinitely to inform other systems coming on line. If it persists, it must not send them more than once every 10 minutes (after the initial startup sequence). If the RAP process sends polls as part of its startup, it must not persist in sending them after the startup sequence.

The command code for no-operation is always 0, regardless of RAP version.

2.2.2 Poll

A poll command packet requests that the other side transmit either a no-operation packet, or some other packet if sent without delay. (i.e., receivers MUST NOT delay a response to a poll by waiting for some other packet expected to be queued soon.)

The poll command code is always 1, regardless of version, and the length is always 8.

Each RAP implementation runs a timer for each connection, to ensure that if the other system becomes unreachable, the connection will be closed or reset. The timers run at each end of the connection are independent; each system is responsible for sending polls in time to reset its own timer.

The timer MUST be reset (restarted) on the receipt of any RAP packet, regardless of whether the version or command code is known.

In normal operation, if route updates are being sent in both directions, polls may not be necessary for long periods of time as the timers are continually reset. When the connection is quiescent, both timers will typically get reset as a result of the side with the shorter timer doing a poll, and then getting a no-operation in response. RAP implementations MUST NOT be dependent in any way on the size or existence of the remote timer.

An implementation that has access to information from the TCP layer, such as the results of TCP layer keepalives, MAY use this instead of or in addition to a timer. However, the use of TCP keepalives is discouraged, and this procedure does not ensure that the remote RAP process is alive, only that its TCP is accepting data. Thus a failure mode exists that would not exist for active RAP layer polls.

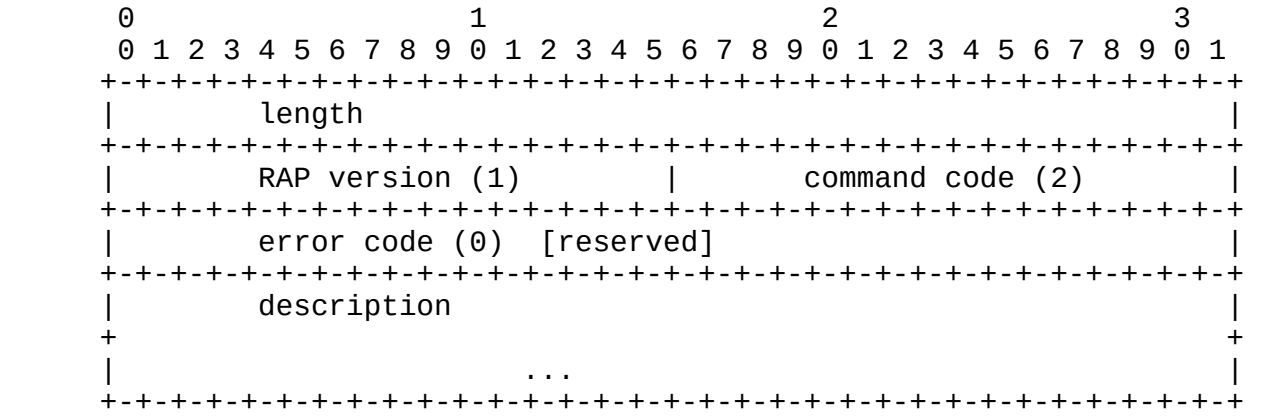
The timer MUST be implemented, SHOULD be configurable in at least the range 1 to 10 minutes on a per-peer basis, and MAY be infinite (disabled) by explicit configuration.

On UDP, a system (router or non-routing host) may send RAP polls to attempt to locate candidate peers or possible gateways. Such a system must not persist in sending polls after its startup sequence, except that a system which actually has offered traffic for non-local destinations, and has no available gateways, may continue to send periodic polls to attempt to acquire a gateway.

2.2.3 Error

The error packet is used to report an error, whether fatal, serious or informational. It includes a null terminated text description in ISO-10646-UTF-1 of the condition, which may be useful to a human administrator, and SHOULD be written to a log file. (The machine is not expected to understand the text.)

Errors are actual failures (in the interpretation of the receiver) to use the correct syntax and semantics of the RAP protocol itself, or "failure" of the receiver to implement a version of the protocol. Other conditions that may require action on the part of the peer (such as purging a route) are given their own command codes.

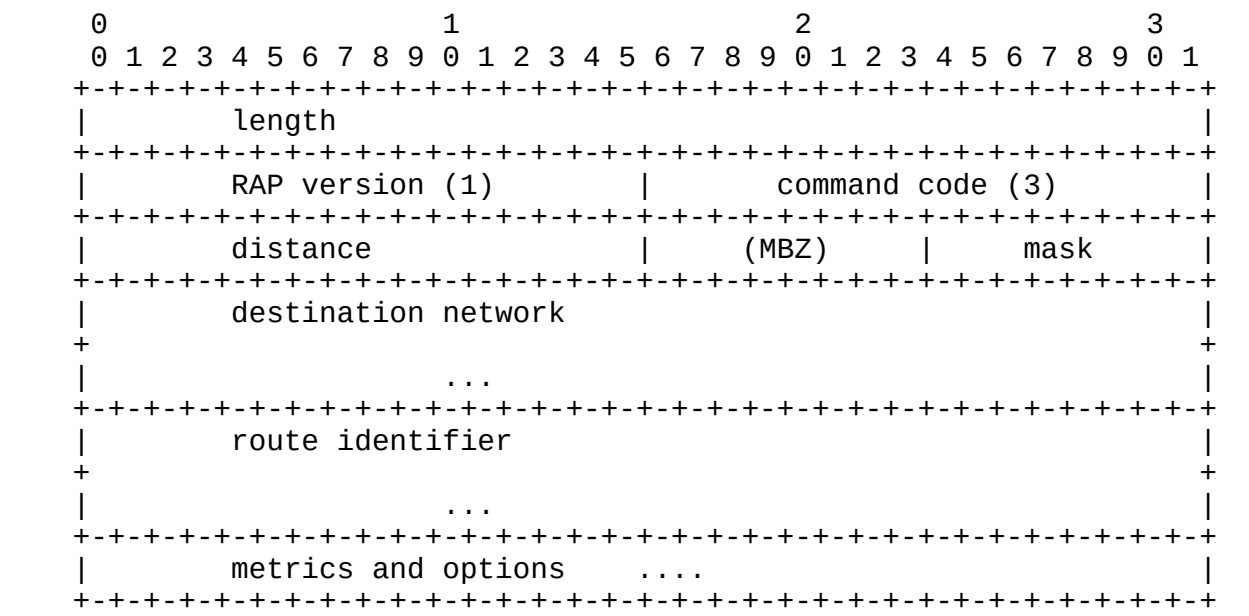


The RAP system receiving an Error packet MUST NOT regard it as fatal, and close the connection or discard routes. If the sending system desires the condition to be fatal (unrecoverable), its proper action is to close the connection. This requirement is to prevent the kind of failure mode demonstrated by hosts that killed off TCP connections on the receipt of ICMP Host-Unreachable notifications, even when the condition is transient. We do not want to discourage the reporting of errors, in the way that some implementations avoided sending ICMP datagrams to deal with overly sensitive hosts.

An error packet MUST NOT be sent in response to something that is (or might be) an error packet itself. Subsequent versions of RAP should keep the command code point (2) of the error packet.

2.2.4 Add Route

The add route command offers a route to the receiving peer. As noted later, it MUST be a route actually loaded into the forwarding database of the offering peer at the time the add route is sent.



The add route command describes a single offered route, with the metrics and other options (such as policies) associated with the route.

Distance is a simple count of the hops to the RAP process (or other routing process) that originated the route, incremented every time the route is forwarded. Its initial value may be greater than 1, particularily for a route that is administratively configured to aggregate routes for a large network or AD. It may also enter the RAP routing domain for the first time with a non-zero distance because the route originated in RIP, OSPF, or BGP; if so, the distance carried in that protocol is copied into the RAP route.

The mask is a count of the number of bits of prefix ones in the binary representation of the network mask. Non-contiguous masks are not supported directly. (The destination restriction option may be used to give another, non-contiguous, mask; the header mask would then describes the number of contiguous ones.)

The route identifier is a 64 bit value that the IP forwarding module on the sending host can use to rapidly identify the route and the next hop for each incoming datagram. The host receiving the route places this identifier into the forward route ID field of the datagrams being sent to this host.

The route ID is also used to uniquely identify the route in the purge route operation.

2.2.5 Purge Route

The purge route command requires that the receiving peer delete a route from its database if in use, and requires that it revoke that route from any of its peers to whom it has offered the route. This command should preferably be sent before the route is deleted from the sending peer's forwarding database, but this is not (cannot be) required; it should be sent without delay when the route is removed.

The command code is 4. The format is the same as add route without any added metrics or options.

If the route identifier in a purge route command is zero, the command requires the deletion of all routes to the destination previously offered by this peer.

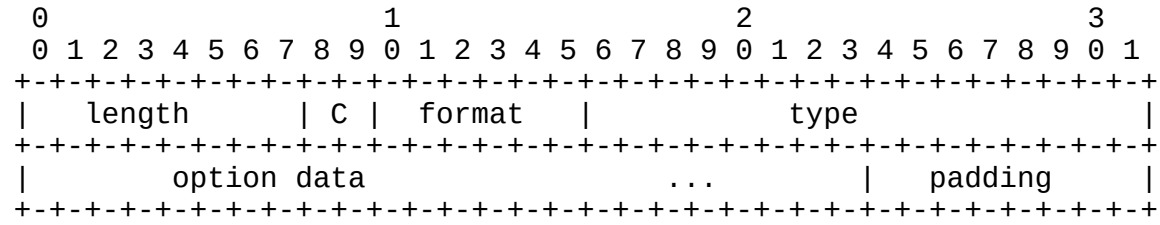
3. Attributes of Routes

There are a rather large number of possible attributes. Possibilities include both metrics, and other options describing for example policy restrictions and alterations of proximity. Any particular route will usefully carry only a few attributes or none at all, particularily on an infrastructure backbone. A reasonable policy for the routers that make up a backbone might be to strip all attributes before propagating routes (discarding routes that carry attributes with class indications prohibiting this), and then adding (for example) an AUP attribute to all routes propagated off of the backbone. A less drastic method would be to simply prefer routes with no restrictions, but still propagate a route with restrictions if no other is available.

Most options can occur more than once in a route if there is any sensible reason to do so.

3.1 Metric and Option Format

Each metric or option for a route begins with a 32 bit header:



RAP Option/Metric Header Format

A description of each field:

length	length of the option or metric
C	option class, see below
format	data format
type	option type identifier
data	variable length

3.1.1 Option Class

This field tells implementations what to do with routes containing options or metrics they do not understand. No implementation is required to implement (i.e., understand) any given option or metric by the RAP specification itself, except for the distance metric in the RAP header.

Classes:

0	use, propagate, and include this option unmodified
1	use, propagate, but do not include this option
2	use this route, but do not propagate it
3	discard this route

Note that class 0 is an imperative: if the route is propagated, the option must be included.

Class and type are entirely orthogonal, different implementations might use different classes for the same option or metric.

3.1.2 Type

The type code identifies the specific option or metric. The codes are part of the option descriptions following.

Type 0 indicates a null (no-operation) option. It should be class zero, but an implementation that "understands" the null option may decline to propagate it.

Note that since an implementation may delete an option of class 1 by simply setting its type to 0 and forwarding the route description, class 1 does not provide any confidentiality of the content of an option.

3.1.3 Format

The format field specifies the format of the data included after the option header. Formats:

0	none, no data present.
1	one or more 32-bit signed integers
2	a character string, null terminated
3	one or more real numbers
4	an octet string
5	one real, followed by a character string

Format is also orthogonal to type, but a particular type is usually only reasonably represented by one format. This allows decoding of all option values for logging and other troubleshooting, even when the option type is unknown. (A new unknown format will still present a problem.)

Format 4, octet string, is to be represented in dotted-decimal byte form when printed; it is normally an internet address.

Format 5 is intended for dimensioned parameters with the character string giving the dimension or scale.

3.2 Metrics and Options

As much as possible, metrics are kept in the base units of bytes and seconds, by analogy to the physics systems of MKS (meter-kilogram-second) and CGS (centimeter-gram-second) of base units.

Bytes aren’t the real primitive, the bit is. We are thus using a multiple of 8 that isn’t part of what one would come to expect from a decimal metric system that uses the other prefixes. However, since K (kilo) is often taken to be 1024, and M (mega) to be 1,048,576 (or even 1,024,000) we allow this liberty.

Distance is measured in units also unique to the field. It is the integer number of times that a datagram must be forwarded to reach the destination. (Hop count.)

3.2.1 Distance

The Distance metric counts the number of hops on a route; this is included in the RAP route command header.

The initial distance at insertion into the RAP domain by the origin of the route MUST be less than or equal to 2z, where z is the number of zero bits in the route mask.

If the origin derives the route from RIP or OSPF, and the distance exceeds 2z, the route must not be used.

When a router originates a route designed to permit aggregation, the distance is usefully set to more than 0; this allows simple subset aggregation without propagating small distance changes repeatedly as the internal diameter of the described network changes.

For example, for routers designated to announce a default route for an AD, with a 24/48 mask, the maximum initial distance is 96.

3.2.2 Delay

The Delay metric (Type = 2) measures the one-way path delay. It is usually the sum of delays configured for the gateways and interfaces, but might also include path segments that are actually measured.

Format is real (3), with one value. The units are seconds.

3.2.3 MTU

The MTU metric (Type = 3) measures the minimum value over the route of the Maximum Transmission Unit, i.e., the largest IP datagram that can be routed without resulting in fragmentation.

Format is one integer, measuring the MTU in bytes.

3.2.4 Bandwidth

The Bandwidth metric (Type = 4) measures the minimum bandwidth of the path segments that make up the route.

Format is one real, representing bandwidth in bytes/second.

3.2.5 Origin

The origin attribute (type = 5) identifies the router that originally inserted the route into the RAP domain. It is one of the IP addresses of the router, format is 4.

3.2.6 Target

The target attribute (type = 6) identifies a host or network toward which the route should be propagated, regardless of proximity filtering that would otherwise occur. This aids in the establishment of tunnels for hosts or subnets "away from home." It can be used to force the route to propagate all the way to the home network, or to try to propagate a better route to a host that the origin has established a connection (e.g., TCP) with. Note that a router can distinguish these two cases during proximity filtering by comparing the route described with the host or network identified by the target option.

Format is 4.

3.2.7 Packet Cost

The packet cost metric (type = 7) measures the actual cost (to someone) of sending data over the route. It is probably either class 3 or 0. Format is 5.

The real number is the cost in currency units/byte. Tariffs set in packets or "segments" should be converted using the nominal (or actual path) size. For example, Sprintnet charges for DAF connections within its network are US$1.40/Ksegment thus for segments of 64 bytes, the cost is 0.000021875 USD.

The string is the 3 capital letter ISO code [ISO4217] for the currency used. Funds codes and codes XAU, XBA, XBB, XBC, XBD, and XXX are not used.

If a route already has a packet cost in a different currency associated with it, another instance of this option should be added. RAP implementations MUST NOT attempt to convert the currency units except when actually making a route selection decision. That is, the effects of a currency conversion should never be propagated, except for the proper effect of such a selection decision.

3.2.8 Time Cost

The time cost metric (type = 8) measures the actual cost of holding one or more paths in the route open to send data. It is probably either class 3 or 0. Format is 5.

The real number is the cost in currency units/second. For example, Sprintnet charges for international connections (to typical destinations) are US$10/hour so the cost is 0.002777778 USD.

The other notes re codes used and conversions in the previous section also apply.

3.2.9 Source Restriction

A source restriction option (type 9, format 4, class 2 or 3) indicates that a route may only be used by datagrams from a particular source or set of sources. The data consists of a network or host number, and a mask to qualify it. If multiple source restriction options are included, the restriction is the logical union of the sources specified; i.e., any are permitted.

Source restrictions must be added to routes when the RAP system has security filters set in the IP forwarding layer. This is necessary to prevent datagrams from taking "better" routes that end in the datagram being silently discarded at the filter. Note that this propagates confidential information about the security configuration, but only toward the net authorized to use the route if the RAP implementation is careful about where it is propagated.

3.2.10 Destination Restriction

A destination restriction option (type 10, format 4, class 3) serves only to provide a non-contiguous mask, the destination already having been specified in the command header. Data is the destination network and mask.

3.2.11 Trace

Trace (type 11, format 4, class 0) provides an indication that the route has propagated through a particular system. This can be used for loop detection, as well as various methods of troubleshooting. The data is one internet address, one of the addresses of the system. If an arriving route already carries a trace identifying this system, and is not an update, it is discarded. If it is an update, the route is purged.

Trace SHOULD NOT be simply added to every route traversing a system. Rather, it should be added (if being used for loop detection) when there is a suspicion that a loop has formed.

When the distance to a destination has increased twice in a row in a fairly short period of time, and the number of trace options present in the route did not increase as a result of the last update, the RAP process should add a trace option identifying itself to the route. Effectively, when a loop forms, one router will select itself to be a tracer, adding itself and breaking the loop after one more turn. If that fails for some reason, another router will add its trace. Each

router thus depends in the end only on its own trace and will break the loop, even if the other routers are using other methods, or simply counting-out the route.

3.2.12 AUP

The AUP (Acceptable Use Policy) option (type 12, format 2, class any), tags a route as being useable only according to the policy of a network. This may be used to avoid traversal of the net by (for example) commercial traffic, or to prevent un-intentional use of an organization's internal net. (It does not provide a security barrier in the sense of forwarding filters, but does provide cooperative exchange of information on the useability of a net.)

The data is a domain name, probably the name of the network, although it may be the name of another organization. E.g., the routers that are subject to the NSF AUP might add NSF.NET as the descriptor of that policy.

3.2.13 Public

Public (type 13, format 0, class 2 or 3) marks the route as consisting in part of a public broadcast medium. Examples of a public medium are direct radio broadcast or a multi-drop cable in which other receivers, not associated with the destination may read the traffic. I.e., a TV cable is a public medium, a LAN within an organization is not, even if it can be easily wiretapped.

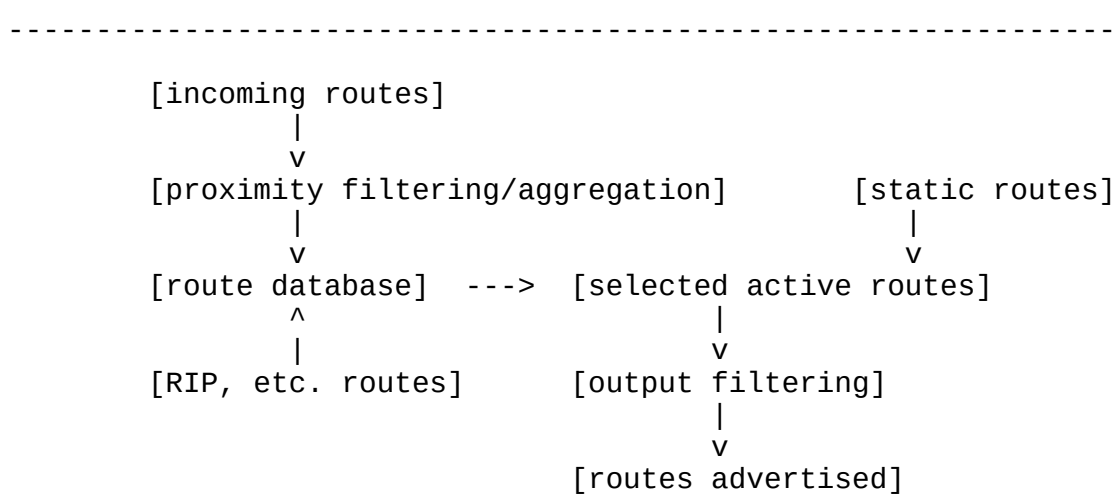
This is intended for use by cable TV providers to identify routes that should not be used for private communications, in spite of the attractively high bandwidth being offered.

4. Procedure

Routing information arrives in the RAP process from other peers, from (local) static route and interface configuration, and from other protocols (e.g., RIP). The RAP process filters out routes that are of no interest (too detailed or too "far away" in the topology) and builds an internal database of available routes.

From this database, it selects routes that are to be active and loads them into the IP forwarding database.

It then advertises those routes to its peers, at a greater distance.



4.1 Receiver filtering

The first step is to filter out offered routes that are too "far away" or too specific. The filter consists of a maximum distance at which a route is considered usable for each possible (contiguous) mask.

Routers that need universal connectivity must either pass through the filter all routes regardless of distance (short of "infinity"), and use aggregation to reduce them, or have a default route to a router that does this.

The filter may be adjusted dynamically to fit limited resources, but if the filter is opened, i.e., made less restrictive, there may be routes that have already been offered and discarded that will never become available.

4.2 Update of metrics and options

The process then updates any metrics present on the route to reflect the path to the RAP peer. MTU and bandwidth are minimized, delay and cost are added in. Distance is incremented. Any unknown options cause class-dependent processing: discarding the option (class 2) or route (3), or marking the route as non-propagatable (1).

Policy options that are known may cause the route to be discarded at this stage.

4.3 Aggregation

The next step is to aggregate routes that are subsetted by other routes through the same peer. This should not be done automatically in every possible case. The more information that is propagated, the more effective the use of forward route identifiers is likely to be, particularily in the case of aggregating into a default route.

In general, a route can be included in an aggregate, and not propagated further, if it is through the same peer (next hop) and has a smaller distance metric than the containing route. (Thus datagrams will always travel "downhill" as they take more specific routes.)

The usual case of aggregation is that routes derived from interface configurations on the routers from which they originated are subsumed into routes offered by routers explicitly configured to route for an entire network, area, or AD. If the larger area becomes partitioned, unaggregatable routes will appear (as routes outside the area become the shortest distance routes) and traffic will flow around the partition.

Attributes of routes, particularily policy options, may prevent aggregation and may result in routes simply being discarded.

Some information about aggregation also needs to be represented in the forwarding database, if the route is made active: the router will need to make a decision as to which forward route identifier to use for each datagram arriving on the active route.

4.4 Active route selection

The router selects those routes to be entered into the IP forwarding database and actively used to forward datagrams from the set of routes after aggregation, combined with routes derived from other protocols such as RIP. This selection may be made on any combination of attributes and options desired by local policy.

4.5 Transmitter filtering

Finally, the RAP process must decide which routes to offer to its peers. These must be a subset of the active routes, and may in turn be a selected subset for each peer. Arbitrary local policies may be used in deciding whether or not to offer any particular route to a given peer.

However, the transmitter must ensure that any datagram filters are represented in the offered route, so that the peer (and its peers) will not route into a black hole.

4.6 Last resort loop prevention

RAP is designed to support many different kinds of routing selection algorithms, and allow them to interact to varying extents. Routes can be shared among administrations, and between systems managed with more or less sophistication.

This leaves one absolute requirement: routing loops must be self-healing, regardless of the algorithm used on each host. There are two caveats:

1. A loop will not fix itself in the presence of an error that continually recurs (thus re-generating the loop)

2. The last resort algorithm does not provide rapid breaking of loops, only eventual breaking of them even in the absence of any intervention by (human) intelligence.

The algorithm relies on the distance in the RAP route header. This count must be updated (i.e., incremented by one) at each router forwarding the route.

Routers must also impose some limit on the number of hops permitted in incoming routes, discarding any routes that exceed the limit. This limit is "infinity" in the classic algorithm. In RIP, infinity is 15, much too low for general inter-domain routing.

In RAP, infinity is defined as 2z + i, where z is the number of zero bits in the mask (as described previously) and i is a small number which MUST be configurable.

Note that RAP depends on the last resort algorithm, "counting to infinity," much less than predecessors such as RIP. Routes in the RAP domain will usually be purged from the net as the purge route command is flooded without the delays typical of periodic broadcast algorithms. Only in some cases will loops form, and they will be counted out as fast as the routing processes can exchange the information.

5. Conclusion

Unlike prior routing protocols, RAP is designed to solve the entire problem, from hands-off autoconfiguration of LAN networks, to implementing the most complex policies of international carriers. It provides a scaleable solution to carry the Internet forward to a future in which essentially all users of data transmission use IP as the fabric of their networks.

6. Appendix: Real Number Representation

Real numbers are represented by a one byte exponent, e, in excess-128 notation, and a fraction, f, in excess-8388608 notation, with the radix point at the right. (I.e., the "fraction" is actually an integer.)

e is thus in the range 0 to 255, representing exponents (powers of 2) in the range 2^-128 to 2^127.

f is in the range 0 to 16777215, representing numbers from -8388608 to 8388607

The value is (f-8338608) x 2^(e-128)

The real number is not necessarily normalized, but a normalized representation will, of course, provide more accuracy for numbers not exactly representable.

Example code, in C:

```
#include <math.h>

typedef struct {
        unsigned e : 8;
        unsigned f : 24;
        } real;

double a;          /* input value */
real r;
double b;          /* output value */
double d;
int e32;

/* convert to real: */

d = frexp(a, &e32);
r.e = e32+105;
r.f = (int)(d*8388608.0) + 8388608;

/* convert back: */

b = ldexp((double)r.f - 8388608.0, (int)r.e - 128);
```

7. References

[ISO3166] International Organization for Standardization. Codes for the Representation of Names of Countries. ISO 3166, ISO, 1988.

[ISO4217] International Organization for Standardization. Codes for the representation of currencies and funds. ISO 4217, ISO, 1981.

[RFC791] Postel, J., "Internet Protocol - DARPA Internet Program Protocol Specification", STD 5, RFC 791, DARPA, September 1981.

[RFC1058] Hedrick, C., "Routing Information Protocol", STD 34, RFC 1058, Rutgers University, June 1988.

[RFC1247] Moy, J., "OSPF Version 2", RFC 1247, Proteon, Inc., July 1991.

[RFC1287] Clark, D., Chapin, L., Cerf, V., Braden, R., and R. Hobby, "Towards the Future Internet Architecture", RFC 1287, MIT, BBN, CNRI, ISI, UCDavis, December 1991.

[RFC1338] Fuller, V., Li, T., Yu, J., and K. Varadhan, "Supernetting: an Address Assignment and Aggregation Strategy", RFC 1338, BARRNet, cicso, Merit, OARnet, June 1992.

[RFC1475] Ullmann, R., "TP/IX: The Next Internet", RFC 1475, Process Software Corporation, June 1993.

8. Security Considerations

Security issues are discussed in sections 3.2.9 and 3.2.12.

9. Author’s Address

Robert Ullmann
Process Software Corporation
959 Concord Street
Framingham, MA 01701
USA

Phone: +1 508 879 6994 x226
Email: Ariel@Process.COM