Robert L. Ullmann
Prime Computer, Inc.
December 1990

TP/IX: The Next Internet

Status of this memo

This memo describes some ideas for the next version of the Internet protocols.

It is not even experimental, just so much blue sky at this time.

It is cast as a protocol description to provide the rigor of a formal description, not to indicate that the details are in some way immutable. The author feels that he has read too many descriptions of possible methods (in various areas, not just network communications) that consist mostly of hand-waving.

Discussion of the TP/IX protocols is on list `TP-IX@List.Prime.COM`, to subscribe, send a message to `TP-IX-Subscribe`; to cancel, send to `TP-IX-Cancel`. All other requests or problems to `TP-IX-Request`.

Distribution of this memo is unlimited.

Table of Contents

List of Figures

List of Tables

1. Introduction

The following are some of the objectives and considerations used in the design of the TP/IX protocol.

Note that "Internet" is used to refer to the actual connected network called Internet, and "internet" as a generic name for the protocols.

- It must interoperate with the existing internet protocols. In particular, it must support existing applications without modifications, except where those applications explicitly reference transport and internetwork layer parameters.
- It must run on any physical and link layer, with appropriate encapsulation. In particular, it must be capable of using the IP Internet [11]. This is essential to early testing, allowing any or all Internet sites to "come out and play".
- Routers should be able to use cut-through routing, forwarding a packet as it arrives. This means the information required must be at the start of the packet, to be used and stripped off, with reverse route information added to the end. (The reader should be familiar with SIRPENT [3].)
- There must be no modifications to the content of the network packet as it is routed, as optical switches may not be able to do it, and electronic switching will be faster if it does not have to. This means no decrementing of time-to-live, and no checksum recomputations.
- Packets may be of any size, including many megabytes.
- Bandwidth may be anywhere from 100bps to 1Tbps or more.
- The network must scale easily to tens of millions of subnets, and thousands of millions of users.
- Networks should configure themselves, at least within a local area. (Local topologically, not necessarily physically.) The next generation of users is not going to know, or want to know, how to administer a local network; they have other work to do. Configuration should consist of naming your machine in the domain space, and plugging it in.
- At the other extreme, the network layer must not prevent applications from doing an arbitrary amount of policy routing, bandwidth preallocation, planned or static routing, and so on.[1]
- Applications supported include real time high-density video, CD quality sound, Internet-wide file system, as well as the more familiar remote session access, file transfer, and mail.

[1]Investigation of this very open-ended subject has a long way to go.

1.1. Philosophy

Protocols should become simpler as they evolve.

A protocol should have a simple core function, with all other features and tweaks expressed as options. This enables very simple implementations (often the key to very high performance), with only the additional features added that are in fact useful.

True interoperability is much easier to attain with a simple core protocol; it may be the difference between reality and impossibility.

2. IX: Internet datagram service

Each IX datagram in transmission consists of 1 or more path descriptors specifying the route the datagram is to follow, then the next layer (transport) protocol segment, and then 1 or more path descriptors describing the origin of the datagram, the route traversed thus far.

0 7	8 15	16 23	24 31
forward route			
transport layer segment			
reverse route			

Figure 1: IX datagram format

In the following description, the term "router" is used to indicate a particular component of every node in the TP/IX internet, whether it is a host in the normal sense or a dedicated router.

As the datagram moves through the network, each router interprets the first path descriptor as the interface and address to use on that interface in forwarding the datagram. In the process, it removes the descriptor from the datagram.

The descriptor consists of a field specifying its length (in multiples of 32 bits), a one byte identifier of the path, and the address information to use on that path.

0 7	8 15	16 23	24 31
length	path	address	
address ...			

Figure 2: Path descriptor: forward route

The length may be any value from 1 to 254 inclusive, representing lengths in the range 4 to 1020 bytes.

The path descriptor is any value from 0 to 255. Path 0 has a defined meaning: it is the destination host name, encoded as in appendix C. It indicates that the 32 bit word following is the start of the transport layer header, with the transport identifier in the low 16 bits.

Other than path 0, the meanings of the path identifiers and the structure and syntax of the address information are specific to the router. Not just the implementation, but the particular instance as well: descriptors are created by the router that will be interpreting them (or possibly by a cooperating host or service), and are therefore specific to that instance.

A particular router might use its own interface numbers for the path, and (for example) with interface 2 being an ethernet, use the MAC layer address as the rest of the descriptor.

When a datagram is received that begins with a path 0 descriptor, the router starts by comparing the host name with its own name (or aliases). If it matches, the datagram is passed to the appropriate transport layer protocol.

If the host name does not match, the router uses information available from the RAP process (see 5) to route the datagram. If none is available, it may initiate a DNS request, forward the datagram on a default route, or both. It might also start an ARP request on any attached broadcast network, in which case it may hold the datagram for a very short time[2] to see if an ARP reply is received.

If information from RAP was used, the router also passes a message to the local RAP process, requesting that it explicitly propagate the route back to the (adjacent) host from which the IX datagram was received.

As a datagram leaves the router on an interface, the router adds to the end a reverse path descriptor, indicating the path back through the router to the source. If the datagram originated on this host, it ensure that it ends with the proper path 0 descriptor for the transport. The address in the trailing path 0 descriptor must always be the canonical name for the host, except when the transport is replying to an incoming datagram addressed to an alias.

0 7	8 15	16 23	24 31
address information ...			
...		path	length

Figure 3: Path descriptor: reverse route

The trailing descriptor ends with the path identifier and the length.

If the datagram is being cut-through routed, and a receive error occurs, or if a tranmission must be pre-empted, the transmission in progress is ended by sending all-ones bytes, i.e. a "length" of 255 for the last descriptor. It must send 1 to 4 bytes as required to end on a 32 bit boundary. The receiver discards the datagram. If it is also doing cut-through, it simply stops at the end of the received data, and the next router in sequence will see the abort termination.

2.1. IP encapsulation

IX datagrams are transmitted on IP as IP protocol 35.[3] The IX datagram immediately follows the IP header. The IP total length is used to determine the end of the IX datagram.

The IX MTU is assumed to be large enough to provide 1024 bytes of application data. Specific implementations may want to try to use RAP to set MTU lower to avoid IP fragmentation, but in the normal case IP fragmentation will simply be allowed to occur.

An IP MTU of 1500 bytes, as provided by ethernet and most HDLC implementations, will usually allow avoiding fragmentation.

---

[2]A good limit might be less than 500 msec.

[3]IP protocol 35 is not registered as of April 1990

2.2. Ethernet, FDDI, token ring encapsulation

On ethernet, and on compatible MAC-layer LANS, IX datagrams are transmitted with type [TBA].

Specific implementations may of course use any local method desired, but will not be able to interoperate with other implementations.

2.3. X.25 encapsulation

On X.25, the IX datagram is transmitted as a single L3 packet or as a more data sequence with total length of at least 1600 bytes; specific connections may use RAP to announce larger sizes.

The virtual circuits are established with a protocol ID of C0F00001 (hex, is 192 240 0 1 decimal). Other values of the third and fourth octets are available for other internet protocols (65534 values), including a possible compressed form of IX.[4].

2.4. ISDN encapsulation

On ISDN, the Q.931 [2] user-user setup element is used to specify an X.244 encoded protocol identifier. The X.244 code is C0F00001, as for the X.25 encapsulation.

The lower layer bearer capability may be selected by the router attempting to establish the call; it should be prepared to respond to releases specifying diagnostics indicating that the remote cannot handle the capability by attempting a call again with different (presumably "lower") capability.

It must at least support, and eventually default to:

```
(refer to Q.931 table 4-18)

coding standard:        CCITT
transfer capability:    unrestricted digital
transfer mode:          packet
user layer 1:           rate adaptation
user layer 3:           X.25
```

Table 1: Q.931 Setup Parameters

This permits interoperation between ISDN hosts and PSDN hosts operating on the existing X.25 networks. Calls that are entirely within ISDN may be attempted with other parameters. In either case the X.244/C0F00001 user data is used to identify the call as IX internet.

[4] Like Van Jacobson IP compression, which might have its own C0F0.... protocol ID

3. TP: Transport protocol

3.1. TP segment format

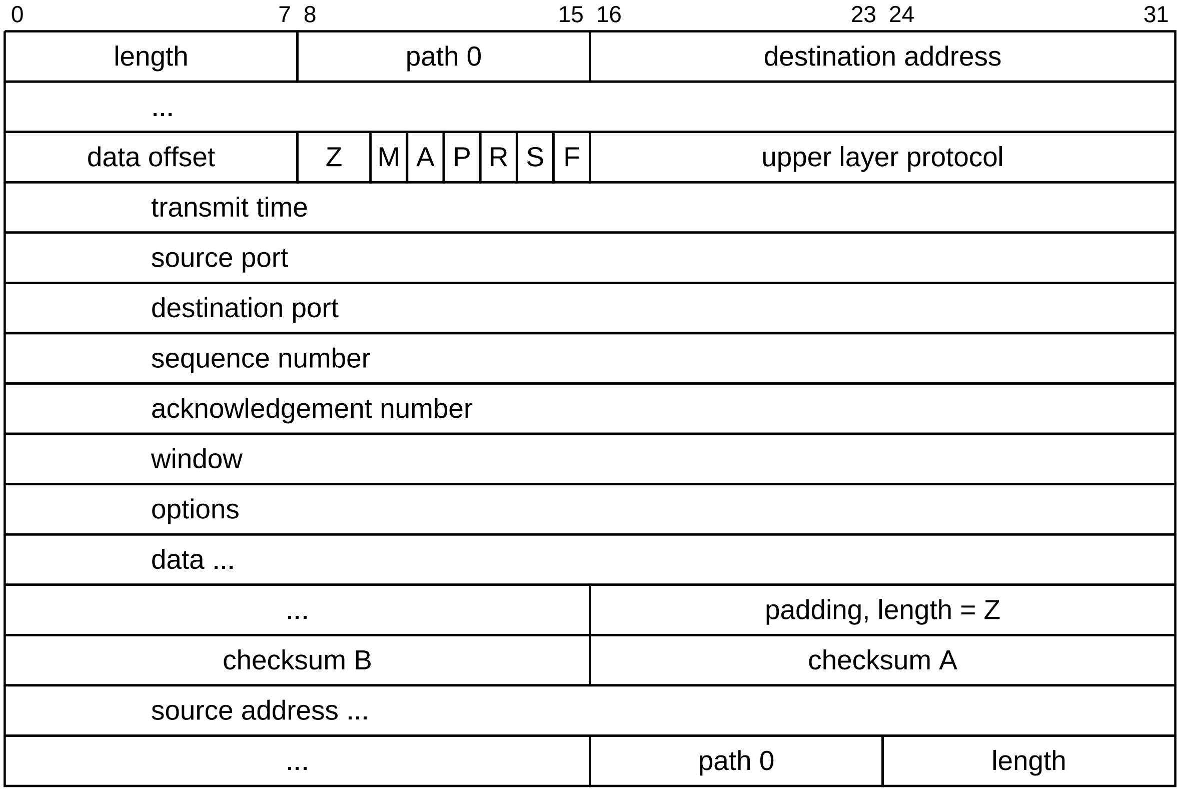

Figure 4: TP Segment Format

This diagram represents a complete IX packet, with the path 0 transport descriptors, but without any network path descriptors.

A description of each field:

data offset	offset, in 32 bit words, from this point to the first data byte
Z	two bit unsigned count of the number of pad bytes added to the data section to fill out to a 32 bit boundary.
M	More data. When set, this segment is an incomplete application record. When reset (0) this segment completes a record, or records are not being used. (While record boundaries are generally not a good idea, some protocols, such as ISO/OSI applications, depend on them.) The TP layer does not control the size of complete records.

A	Acknowledgement. When set, the acknowledgement field is valid.
P	Push. Data must be pushed through, not buffered by receiving TP. An example is the end of a application "turn".
R	Reset. Request to reset (clear) the TP layer connection.
S	Synchronize. (Syn) Start connection, inititalize from this sequence number.
F	Finish. (Fin) Controlled close of connection.
upper layer protocol	TP is protocol 6
transmit time	32 bit millisecond time segment was transmitted (see appendix A).
source	TP port at the source address.
destination	TP port at the destination address.
sequence	number of the first data byte in this packet, or the next number to be used if no data.
acknowledgement	number of the next data byte expected, one more than the last byte received.
window	number of bytes forward from the current acknowledgement that may be sent, or (in a Syn/no-Ack) the number allowed initially.
options	variable length.
data	any amount of data, up to the minimum of the max segment size (MSS) option and the route MTU less path descriptors. If no option has been received, up to 1024 bytes.
data padding	0 to 3 bytes (not necessarily 0, but included in the checksum), Z is the number of data pad bytes.
checksum A, B	is a 16 bit Fletcher checksum [12, 4] of the segment, from the start of the leading path 0 descriptor to the end of the trailing path 0 descriptor.

3.2. Port numbers

Port numbers are divided into several ranges: (all numbers are decimal)

0	reserved
1-65535	internet registered ("well-known") protocols
65536	reserved
66537 up	dynamic assignment

Table 2: TP Port Numbers

3.3. TP options

Each TP option has a 32 bit header:

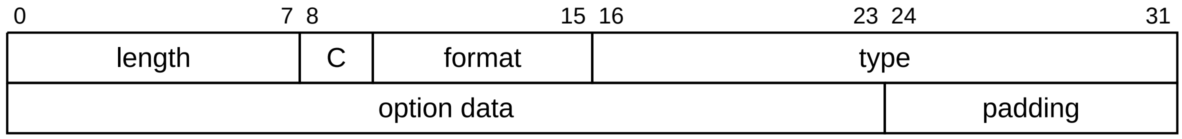

Figure 5: TP Option Format

A description of each field:

length	length of the option or metric.
C	option class, reserved, MBZ.
format	data format, as given in table 3.
option data	variable length.

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 (see appendix B)
4	internet name string (see appendix C)
5	one real, followed by a character string

Table 3: Option Data Formats

3.3.1. Maximum Segment Size

The MSS option (type = 1) specifies the maximum segment size that the sending TP is prepared to recieve, or select an optimum smaller size.

The size is an integer, the maximum number of bytes.

3.3.2. Urgent Pointer

The urgent pointer (type = 2) is a single unsigned number in the sender's sequence space, normally within or adjacent to the segment carrying the option.

3.3.3. Selective Ack

The selective acknowledgement (type = 3) acknowledges receipt of a particular segment or segments, not contiguous with the edge of the window. [experimental]

Data is two unsigned numbers: the first sequence number of the data to be acknowledged, and the first following that is not acknowledged (i.e. last acked plus one).

It may be unreasonable to expect the remote TP to do anything with a selective ack unless it corresponds exactly to a segment in the data stream. This option does not accelerate retransmission of un-acknowledged segments; see the negative-acknowledgemnet option.

3.3.4. Negative Acknowledgement

The negative acknowledgement option (type = 4) requests the immediate retransmission of a segment. [experimental]

Data is a single sequence number, the number of the first byte of the segment to be retransmitted.

Both selective and negative acknowledgements may be sent as options on a segment (typically the segment will be a normal acknowledgement), and there may be more than one of either or both options.[5]

3.3.5. Flow Rate

This option (type = 5) requests that the remote TP throttle flow-rate to the specified number of Kbytes/second. (format is integer or real) [experimental]

This is not used for network layer flow control; this is to provide for an transport layer control of the remote application process if provided by the local and remote TP. The intent is to allow the possibility of transport rate control when the network is faster than the (remote) application.

This option may be useful on high-bandwidth long-delay network paths.

---

[5] This is a area open to experimentation, especially on different types of network.

4. UDP: User datagram protocol

4.1. UDP datagram format

<table>
<tr><th>0 7</th><th colspan="3">8 15</th><th>16 23</th><th>24 31</th></tr>
<tr><td>length</td><td colspan="3">path 0</td><td colspan="2">destination address</td></tr>
<tr><td colspan="6">...</td></tr>
<tr><td>data offset</td><td>Z</td><td>C</td><td>MBZ</td><td colspan="2">upper layer protocol</td></tr>
<tr><td colspan="6">transmit time</td></tr>
<tr><td colspan="6">source port</td></tr>
<tr><td colspan="6">destination port</td></tr>
<tr><td colspan="6">options</td></tr>
<tr><td colspan="6">data ...</td></tr>
<tr><td colspan="4">...</td><td colspan="2">padding, length = Z</td></tr>
<tr><td colspan="4">checksum B</td><td colspan="2">checksum A</td></tr>
<tr><td colspan="6">source address ...</td></tr>
<tr><td colspan="4">...</td><td>path 0</td><td>length</td></tr>
</table>

Figure 6: UDP Datagram Format

This diagram represents a complete IX packet, with the path 0 transport descriptors, but without any network path descriptors.

A description of each field:

data offset	offset, in 32 bit words, from this point to the first data byte
Z	two bit unsigned count of the number of pad bytes added to the data section to fill out to a 32 bit boundary.
C	checksum supressed indicator. If 0, the checksum is valid. If 1, the checksum field was not computed and must be ignored.
M	Must be zero. Reserved field.
upper layer protocol	UDP is 17
transmit time	32 bit millisecond time segment was transmitted (see appendix A).
source	UDP port at the source address.

destination	UDP port at the destination address.
options	variable length.
data	any amount of data, up to the route MTU less path descriptors. If no MTU is known, up to 1024 bytes.
data padding	0 to 3 bytes (not necessarily 0, but included in the checksum), Z is the number of data pad bytes.
checksum A, B	is a 16 bit Fletcher checksum [12, 4] of the segment, from the start of the leading path 0 descriptor to the end of the trailing path 0 descriptor.

4.2. Port numbers

Port numbers are divided into several ranges, in the same way as TP ports. The well-known number assignments are the same as used for TP. Particular application protocols may use TP, UDP, or both.

4.3. UDP options

UDP option format is that same as for TP.

There are no defined options at ths time.

5. RAP: Route Protocol

This specification covers only the protocol for distributing and updating the route information. It does not attempt to determine the algorithms used by gateways to make routing decisions with the information.

This is a consequence of the current state of the art: the various bits of information that are or may be useful in automatic routing are fairly well understood, but the advantages and failures of different algorithms are understood to varying degrees.

The intent is to provide a common base for the existing routing algorithms, to facilitate further experimentation, and common metrics that can be shared across boundaries between areas using different algorithms where appropriate.

It also provides sufficient definition for implementations operating in simple environments; for example a set of LANs operating at the same bandwidth with a single external connection. (e.g. RIP [5])

5.1. RAP route packet

RAP is a transport-layer protocol, IX type 9. It operates by exchange of routing information between peers. Each RAP packet has the format shown in figure 5.1.

0 7	8 15	16 23	24 31
offset	name length	upper layer protocol	
transmit time			
expiration time			
hops		stratum	
domain name ...			
...			
options ...			
...			
path descriptors			
...			
checksum B		checksum A	

Figure 7: RAP packet format

The fields of the RAP packet are:

offset	number of 32 bit words from this point to the start of the descriptor list for this route
name length	Number of 32 bit words in the name.
upper layer protocol	RAP is 9
transmit time	32 bit millisecond transmit time
expiration time	Millisecond expiration time of this route. This is set by the originator, and either copied unmodified or reduced (moved to an earlier time) when propagated.
hops	Metric describing the number of links between IX routers between the sending RAP peer and the host described by the route. If the sender is the host described, this would be 0; if the sender is a gate on the same LAN as the host, hops would be 1.
stratum	RAP stratum of sending host, see section 5.4.3.

options	RAP route options and metrics for the route, as specified in section 5.3.
descriptors	The route to the described host, as a sequence of path descriptors in the foward-route format. This does not include the path 0 descriptor at the end, and does not include the path from the receiver to the sending RAP peer. If the host is describing itself, this would be null. The number of descriptors is always equal to hops.[6]
checksum A, B	16 bit Fletcher checksum, including the leading and trailing path 0 descriptors. (like the TP segment checksum)

5.2. Metric and Option Format

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

0 – 7	8 – 15		16 – 23	24 – 31
length	C	format	type	
option data				padding

Figure 8: RAP Option/Metric Format

A description of each field:

length	length of the option or metric.
C	option class, see below (5.2.1).
format	data format.
option data	variable length.

5.2.1. 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 Hops metric in the route header.

Note that class 3 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.

[6] Except in the case of hops = -1, as noted in section 5.3.1.

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

Table 4: RAP Classes

5.2.2. Format

The format field specifies the format of the data included after the option header. The code used are the same as for TP options, see table 3.

Format is also orthogonal to type, but a particular type is usually only reasonably represented by one format.

5.3. Metrics and Options

5.3.1. Hops

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

If the hop count is -1, i.e. all-ones in the field, the route through the peer is to be removed. This is typically done when implementing poison-reverse. In this case, there are no path descriptors given in the route data packet.

5.3.2. 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 paths that are actually measured.

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

5.3.3. MTU

The MTU metric (Type = 3) measures the minimum value over the route of the Maximum Transmission Unit, i.e. the largest IX datagram that can be routed.

Format is one integer, measuring the MTU in bytes.

5.3.4. Bandwidth

The Bandwidth metric (Type = 4) measure the minimum bandwidth of the paths that make up the route.

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

5.3.5. [type 5]

5.3.6. Terminal

The terminal option (type = 6) indicates that this route is to be used, but not propagated. It has format 0, and no data present. It must be class 1, and can be implemented by simply ensuring that it is not implemented.

Using class = 0 to mean always discard will work, but does not seem particularily useful.

5.3.7. Tariff

The Tariff metric (type = 7) measures the actual cost of the route. It is probably either class 3 or 0. Format is 5.

The real number is the cost in currency units/Kbyte for K=1024. Tariffs set in packets or "segments" should be converted using the nominal (or actual path) size. For example, TELENET charges are (or were) US$1.40/Ksegment, for segments of 64 bytes, the cost is therefore 0.0224 USD

The string is the 3 capital letter ISO4217 [6] code for the currency used. Funds codes and codes XAU, XBA, XBB, XBC, XBD, and XXX are not used.

The metric is updated where required by converting the currency in the incoming route to the currency configured for the path to the other gate, and adding the results. This is not expected to be very common:

- automatic routing using tariff will probably be within one organization
- an organization operating within one country or currency zone will be using only that currency
- a multi-national organization will probably make an administrative decision to configure using only one or a few currencies. (I.e. perhaps only XEU, USD, and JPY)

5.4. Procedure

RAP operates by exchange of routing information between peers. The input to the RAP procedure is a set of potential peers; the output is a routing database usable by the local transport interface and by the local IX router for routing in the absence of path descriptors.

5.4.1. Peer acquisition

The RAP process build a list of potential peers from the following sources:

- statically configured neighbors

- hosts on directly connected LANS noted from ARP[7]
- hosts identified from DNS requests made to route IX datagrams
- other hosts soliciting a RAP exchange

Each of these is added to a table of candidate or active peers. The entry must include the expiration time of the source information, i.e. the TTL for a DNS response, the ARP cache timeout for an ARP reply, etc. This is used to eventually drop the host from the peer list.

Each entry also should include a last-used time, to be updated when a route through this peer is actually used by the transport or IX processes; this can be used to age and discard unused peers.[8]

5.4.2. Route information exchange

Periodically[9] each RAP process announces to each peer all of the routes it has that the peer might find useful. The details of some of the possible algorithms are described in the RIP protocol description. [5]

In RAP, other factors are also taken into account. Routes are filtered by distance, domain boundaries, and explicit policy configuration. Routes may also be propagated by the default routing procedure (see 2 when they would not otherwise have been.

5.4.3. Strata

Each RAP host maintains a variable identifying the stratum in a hierarchy used for default routing. When an IX datagram cannot be routed by any other means, it is forwarded to an adjacent host at a lower-numbered stratum of the hierarchy.

Hosts at stratum 1 have no default route, and use only DNS and local RAP information to route datagrams with only a path 0 descriptor. A host is on stratum 1 if and only if it meets the following conditions:

- It is directly connected to an X.25 PSDN, or to an ISDN that can connect to, and be connected to from, any PSDN.
- It does not have *any* call restrictions on the X.25 network. In particular, it is not part of a closed user group or any similar administrative restriction. It must be able to receive and accept calls from any X.25 PSDN address.[10]

---

[7] The part of the ARP procedure specifying caching of replies to other hosts' requests, often not implemented in TCP/IP software, is *required* in TP/IX.

[8] Some care must be taken to ensure that the traffic caused by RAP itself does not update this value!

[9] The period to be determined by any factors the implementation wants to take into account.

[10] It does not have to accept reverse-charge calls, of course.

- It must be on the connected IP Internet, and support the IX/IP encapsulation.
- It must have both an A record (IP Internet address), and an X25 record (X.25/X.121 PSDN address) registered in the domain name system.

Most hosts will be initially on stratum 2, by statically configuring a stratum 1 host, or on the next higher strata by being on the same LAN as a stratum 2 host. Eventually, the ISDN connection may become the standard method of connection, and the requirement of an IP address may be dropped.

A host not on stratum 1 sets its strata variable initially to all ones. It then adjusts it to one more than the lowest value of the strata received from its various peers.

5.5. Last resort loop prevention

RAP is designed to support many different kinds of routing 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. Note that the loops referred to are loops within the routing algorithm, as well as default-routing loops of IX datagrams. The classic case of IP route loops does not occur, since there are necessarily a finite number of leading path descriptors in an IX datagram in transmission.

There are two caveats:

- a loop will not fix itself in the presence of an error that continually recurs (thus re-generating the loop)
- 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 hop count in the RAP route header. This count must be updated (i.e. incremented by one) at each gateway forwarding the route.

Gateways must also impose some limit on the number of hops permitted in incoming routes, discarding any routes that exceed the limit. The limit must be configurable; it should be set to something more than the topological diameter of the network as seen from the gateway.

Note that RIP uses this method, with a fixed limit of 15. This is too low for the existing network.

There are known techniques for increasing the power of this method (split horizon, poison reverse, triggered updates), which should be employed where appropriate.

This is not to detract from more powerful methods, only to provide a last resort which all gates must implement in addition to whatever other methods are employed.

6. Domain Name service

In addition to the usual use of the DNS [9, 10] for binding mail destinations to MX records, and domain names to IP addresses, TP/IX uses the DNS to make wide area network route bindings. In this context, the IP-Internet is considered one of the wide area networks.

There are several new RR types defined for use in the IN (Internet) class of the DNS. These include addresses for PSDN and ISDN service, and a new route-binding RR anagolous to the MX RR for mail routing.

6.1. The X25 resource record

The X25 record identifies the PSDN address in the X.121 numbering plan for the domain name. The address is represented as a string of decimal digits, beginning with the 4 digit DNIC (Data Network Identification Code).

For example:

```
Relay.Prime.COM.   IN   X25       311061700956
```

The encoded form is the ASCII representation of the decimal digits, as a `<character-string>`. Type X25 is 19.

6.2. The ISDN resource record

The ISDN record identifies the PSTN (Public Switched Telephone Network) number of the host, together with any DDI (Direct Dial In) and subaddress (SA) as defined by E.163 [1], the ISDN and PSTN numbering plan.

The number is represented as a string of decimal digits, beginning with the E.163 country code and ending with the DDI if any. If a SA is present, it is represented as another string.

For example:

```
Relay.Prime.COM.   IN   ISDN      150887929603217
sh.Prime.COM.      IN   ISDN      150887929603217 004
```

The RR data is the ASCII representation of the decimal digits. It is encoded as one or two `<character-string>`s, i.e. count followed by characters.

In the Q.931 call setup packet, the address is a string of IA5 characters, while the subaddress is th ebinary represention of the number taken as hexadecimal. Type ISDN is 20.

6.3. The RT resource record

The RT resource record provides a route-through binding for hosts that do not have their own direct wide area network addresses. It is used in much the same way as the MX RR.

The format is one (usually small) integer, representing the preference of the route. Smaller numbers indicate more prefered routes. This is followed by the domain name of the host to route through, which is expected to have at least one A, X25, or ISDN record.

For example,

```
    sh.prime.com.       IN   RT   2    Relay.Prime.COM.
                        IN   RT   10   NET.Prime.COM.
    *.prime.com.        IN   RT   90   Relay.Prime.COM.
                        IN   RT   90   gw2.cs.net.
```

When a host is looking up DNS records to attempt to route a datagram, it first looks for RT records for the destination host, which point to hosts with address records (A, X25, ISDN) compatible with the wide area networks available to the host. If it is itself in the set of RT records, it discards any RTs with preferences higher or equal to its own. If there are no (remaining) RTs, it can then use address records of the destination itself.

The RT host (or destination) is then added to the list of RAP peers, and a RAP exchange initiated. If the path is in fact usable, the RAP process will (presently) add a useable route for the datagram. Until that happens, the datagram(s) are default routed. Successive failures should cause RTs of lesser preference (i.e. higher values) to be utilized.

Wild-card RTs are used exactly as are wild-card MXs.

The concrete encoding is identical to the MX RR. Type RT is 21.

A. Millisecond real time clock

The IX datagram is designed not to require modification at any point in the network, other than the removal of paths as they are used, and the addition of the trailers. Since there are necessarily a finite number of paths, infinite loops do not occur. For both of these reasons no time-to-live field exists.

An origination time is used instead. It provides for discarding of old datagrams by a transport protocol, with the destination host defining what is old. (Unlike TTL, in which the source must know or allow for both the route latency and the destination requirements.)

Each transport protocol may (but is not required to) use the transmit time as defined here.

The clock is a millisecond clock, conceptually started on January 1st, 1900, at 00:00:00.000 UTC. This is the same as the NTP [7, 8] time base.

The low-order 32 bits of the unsigned value of this clock are used in the protocols. This "wraps", or overflows, every 49.71 days. Sequence-space (modulus 2^{32}) comparisons are used. Datagrams up to about 10 days old could be recognized as "recent".

A host can set its clock by finding the number of seconds since the 1900 epoch, correcting for the leap-seconds that have been inserted since, multiplying by 1000 discarding the overflow, and adding the current fraction of a second. There were 14 leap seconds added prior to January 1st, 1988.

Note that a number of systems return seconds since January 1st, 1970 00:00 in the local time zone, not UTC: this must be corrected for.

Using NTP is recommended for accurate coordination of clocks. The NTP timestamp is the same as the IX millisecond clock, after proper format conversion: NTP is binary fixed-point in seconds, with 32 bits before the radix point and 32 following; the millisecond clock is fixed point in milliseconds, with 32 bits before and 0 after. For transport protocols requiring better than millisecond accuracy, the direct use of the NTP timestamp is commended.

A robust TP implementation should be prepared to accept SYN segments with a time value that is less accurate, and remember the offset for use in checking incoming segments during the remainder of the session. However, it MUST NOT skew the time in segments it transmits in an attempt to compensate for the remote.

B. 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);
```

C. Internet name coding

When used in a path descriptor, an internet domain name is encoded as explained below. This is only partly to reduce the length; it also provides a case-independent identifier for the host.

Each label in turn is encoded with a bit string, starting with the first character of the top level domain. The resulting strings thus compare in the same sequence as domain names.

For example, the string `Relay.Prime.COM` is encoded in the sequence `C, O, M, *, P, R, I, M, E, *, R, E, L, A, Y`. The result is, in hex, `4DE8 70E6 2987 2D33 F8`.

	0000	A	0011	N	10110
*	0001	B	01000	O	10111
-	00100	C	01001	P	110000
0	0010100	D	0101	Q	110001
1	0010101	E	0110	R	11001
2	00101100	F	011100	S	1101
3	00101101	G	011101	T	11100
4	001011100	H	01111	U	11101
5	001011101	I	1000	V	111100
6	0010111100	J	100100	W	1111010
7	0010111101	K	100101	X	1111011
8	0010111110	L	10011	Y	111110
9	0010111111	M	1010	Z	1111110

Table 5: Domain Name Coding

The first code (4 zero bits) indicates end of string. The final all-ones code is reserved for possible representation of other characters.

Trailing 0 bits are removed. On decoding, trailing 0 bits are assumed as needed. Note that it is not possible to have an occurance of a zero byte within the string, thus with the addition of a final zero byte, the name string can be handled as a normal null-terminated string.

References

[1] International Telegraph and Telephone Consultative Committee.
Numbering Plan for the International Telephone Service.
In *CCITT Recommendations E.163*. IXth Plenary Assembly, Melbourne, 1988.
Fascicle II.2 ("Blue Book").

[2] International Telegraph and Telephone Consultative Committee.
ISDN User-Network Interface Layer 3 Specification for Basic Call Control.
In *CCITT Recommendations Q.931*. IXth Plenary Assembly, Melbourne, 1988.
Fascicle VI.11 ("Blue Book").

[3] David R. Cheriton.
SIRPENT: A High-Performance Internetworking Approach.
CSL-TR 89-393, Stanford University, July, 1989.
SIRPENT is a trademark of Stanford University.

[4] J. Fletcher.
An Arithmetic Checksum for Serial Transmissions.
In *IEEE Transactions on Communication*, pages 247-252. January, 1982.

[5] C. Hedrick.
Routing Information Protocol.
RFC 1058, Internet, June, 1988.

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

[7] D. Mills.
Network Time Protocol (Version 1) Specification and Implementation.
RFC 1059, University of Delaware, July, 1988.

[8] D. Mills.
Network Time Protocol (Version 2) Specification and Implementation.
RFC 1119, University of Delaware, September, 1989.

[9] P. Mockapetris.
Domain Names -- Concepts and Facilities.
RFC 1034, ISI, November, 1987.

[10] P. Mockapetris.
Domain Names -- Implementation and Specification.
RFC 1035, ISI, November, 1987.

[11] Jon Postel, *editor*.
Internet Protocol.
RFC 791, USC Information Sciences Institute, September, 1981.

[12] J. Zweig, C. Partridge.
TCP Alternate Checksum Options.
RFC 1146, UIUC, BBN, March, 1990.

Author's Address

Robert Ullmann 10-30
Prime Computer, Inc.
500 Old Connecticut Path
Framingham, MA 01701

Phone: +1 508 620 2800 x1736

Email: Ariel@Relay.Prime.COM