IEN 30
Section 2.3.3.10

Postel

# GATEWAY ROUTING

## AN IMPLEMENTATION SPECIFICATION

Virginia Strazisar

Radia Perlman

IEN #30

Internet Notebook Section 2.3.3.10

April 11, 1978

## Introduction

In the past few years, ARPA has supported development of several new networks. In order to take advantage of the services available on different networks, some of these networks have been interconnected with gateway machines. This collection of networks and gateways is referred to as the catenet. The gateway machines are responsible for routing traffic among the networks they interconnect. Currently, the gateways use a fixed routing algorithm. A major drawback of this algorithm is that the gateways cannot route traffic around failed networks or gateways. In order to improve performance in the catenet, and especially to provide alternate routing around failed components, we are proposing that the gateways use an adaptive routing algorithm. The routing algorithm proposed is a variation of the ARPANET shortest path routing algorithm. This algorithm has been modified to provide faster response to changes in internet connectivity and to avoid some problems in the current ARPANET strategy.

The first section of this paper provides a brief description of our model of the gateway and catenet as background for the explanation of the routing design chosen. The next section is an explanation of the routing strategy and the motivation for our design decisions. Finally, a more formal specification is included as a guide for implementing the proposed algorithm.

This is a proposal for an initial implementation of adaptive routing in the gateways. Currently, the gateways use static routing tables, thus, they cannot route around failed gateways or networks. The primary purpose for implementing adaptive routing in gateways at this time is to provide a method for routing around failed gateways and networks. The proposed routing algorithm was defined with this goal in mind.

## Issues for Further Study

The purpose of this paper is to define a routing algorithm in sufficient detail that the routing algorithm can be implemented in a gateway. Although the algorithm as specified here can be implemented in the gateways and will provide the capability for routing around failed components, this paper is not intended to discuss or resolve all gateway routing issues. In preparing this paper, several important issues were recognized. Because of their scope and complexity, they are not discussed in detail, although some of them are mentioned below.

One important problem in designing any routing algorithm is determining the vulnerability of the algorithm. Although the operation of the proposed algorithm has been analyzed with respect to the failure of gateways and networks, we cannot prove that gateways that implement the algorithm will route traffic correctly. More importantly, the question remains as to whether there are situations in which traffic from a set of destinations will not be delivered indefinitely. One approach that can be taken in determining the vulnerability of the algorithm is to investigate the possibility of using a program to verify the algorithm. Unfortunately, the vulnerability of the algorithm often becomes apparent only after much experimentation and operational use.

Another issue related to the vulnerability of the routing algorithm is the vulnerability of the implementation. There are several problems in this area. For example, the routing information sent between gateways may be corrupted because of faulty hardware or software in the gateways or networks; a gateway may generate incorrect routing information because of hardware problems; or the routing algorithm may be incorrectly implemented. These problems may be solved in part by checksumming routing information passed between gateways and by

checksumming the routing tables and the code that implements the routing algorithm. The gateways can also perform consistency checks on the routing information to detect anomalous situations, such as one gateway appearing to be on the best path to all destinations.

A problem related to the vulnerability of the algorithm and its implementation is the authenticity of the gateways. It is assumed that gateways will be added to the existing catenet. In a catenet containing more than a few gateways, it is desirable to allow existing gateways to communicate with new gateways without requiring human intervention to inform each gateway of the existence of the new gateway. There must be a method for gateways to authenticate the existence of other gateways in order to prevent any computer in the catenet from sending routing messages to gateways, either intentionally or through an error in generating or transmitting messages.

Another problem to be resolved is the method for determining the status of the networks and gateways in the catenet. Any routing algorithm must rely on a method for determining whether or not components of the catenet can successfully receive and transmit data. This problem can be considered in two closely related steps: First, how is a working versus non-working component defined, and second, given this definition, how do the gateways determine if a particular component is working? For example, a gateway could be defined as working if it could receive and forward a data packet. However, if the gateway discards a given percentage of the traffic it receives, it might be more desirable to consider the gateway as inoperative. If gateways determine that other gateways are working by sending and receiving packets from them and if some of the traffic sent between gateways is lost, then some algorithm based on the number or percent of packets successfully sent and received could be used by gateways to determine the state of other gateways. This is a complex

problem, especially as the reliability of each network and gateway in the catenet may be very different and may need to be measured in different ways. Further, the reliability requirements for different users of the catenet may be vastly different, one user preferring a short delay path, which is slightly less reliable, and another user requiring a highly reliable path despite the cost.

One final problem, which is not addressed in this paper, is the operation of the catenet. Problems in this area include the monitoring of traffic in the catenet, access controls that prohibit use of some networks or gateways for some traffic, and control of the gateways, including the capability for automatic restarting or reloading.

We will continue to study these issues and will suggest modifications to the proposed algorithm to solve these problems. It is assumed that initially the routing algorithm will be implemented in an experimental environment, which will allow for extensive testing and investigation of the remaining problems.

## Model of the Catenet

Gateways are processes that receive packets from one network and forward these packets to their destination on another network. Gateways may be implemented as a set of processes on a general purpose machine, as a set of processes distributed over two or more machines connected to different networks, or as a set of processes on a dedicated machine. For purposes of describing the routing algorithm, it is assumed that the gateway is implemented on a dedicated machine connected to two or more networks. A gateway machine is connected to networks as a host machine on each network rather than as a part of the network's communications subnet.

Hosts that communicate with hosts on different networks implement an end-to-end internet protocol.(1) The hosts may also provide reliable communications by implementing additional end-to-end protocols based on the internet protocol. An example of such a protocol is the Transmission Control Protocol.(2) The internet protocol is invisible to the local networks of each host. The internet protocol defines an internet header that is contained in the data portion of a local network packet. The internet header contains, as a minimum, a source and destination internet address. These addresses are unique throughout the catenet and specify a network address plus whatever addressing information is needed to deliver the packet within the local network, e.g., a host address. All packets sent through gateways contain this internet header.

The gateways use the information in the internet header to forward traffic to the internet destination as follows. Each gateway removes the local network header from a packet that it has received. The gateway reads the internet destination network address in the packet. If the gateway is physically attached to that network, the gateway constructs a local header for the destination network using the internet destination address to determine the destination address within the network. The gateway then sends the packet through the network to its destination. If the gateway is not attached to the destination network, it forwards the packet to another gateway, G. The gateway uses G's network and host address to compose a local header for the appropriate network and sends the packet through that network to G. Note that the gateway determines where to

---

(1) The internet protocol used will be the protocol defined by the Internet Working Group. A draft of this is available in Jonathan B. Postel, "Draft Internetwork Protocol Specification", Version 2, Information Sciences Institute, University of Southern California, Feb. 1978.

(2) Vinton Cerf and Jon Postel, "Specification of Internet Transmission Control Program," Version 3, Jan. 1978.

send each packet based only on its routing tables and on the packet's internet address. The gateways do not need to maintain state information on end-to-end connections; thus, it is possible to change packet routes dynamically to avoid failed components and to split the traffic load to provide higher bandwidth.

In designing an algorithm to be used in routing packets through the catenet, the size of the catenet is important. As the catenet is continually changing, it is difficult to estimate its size, but some observations can be made. Several factors are contributing to the expansion of the catenet. Local networks are being developed to provide better local computing facilities, and these local networks are being interconnected with larger networks to take advantage of the facilities available in the catenet. For example, local networks, such as the BBN Research Computing Center, and the MIT LCS network, have recently been or will be developed and interconnected with the ARPANET. In addition, special purpose experimental networks are being developed in order to provide facilities such as satellite communications or mobile terminal access. These networks are connected to existing networks to provide access to large computing facilities. Examples of this are the Atlantic Satellite Network and the Packet Radio Network, both of which are connected to the ARPANET. The trends that contribute to the growth of the catenet are offset by the cost in time and money in developing a new network, which is much greater than the cost of adding a host or gateway to an existing network.

We can expect the current catenet to expand, but because of the great expense in developing new networks, we expect the number of networks to remain in the same order of magnitude, i.e., less than 100 networks in the catenet. The number of gateways in the catenet is related to the number of networks. To ensure reliability, major networks will generally be connected via more than one gateway. For simplicity and throughput, gateways are

connected to a small number of networks; most existing gateways are connected to two networks. Thus, the number of gateways in the catenet will be one or two times the number of networks. These approximations have served as guidelines in designing the proposed routing algorithm. The routing algorithm will work with larger or smaller catenets than the one described above. However, if we had considered very small catenets, less than 10 nets, or very large catenets, greater than 100 nets, we might have developed a very different routing strategy.

## The Routing Algorithm

Several methods have been proposed for designing and describing routing algorithms. The proposed routing algorithm is explained in terms of the control mechanism, the decision process, the updating process, and the traffic assignment process.(3) The control mechanism is concerned with where control of the routing process resides, i.e., is it in one machine, distributed across several machines, etc. The decision process involves the choice of a particular route to a destination. The updating process involves updating of routing information; it can be defined in terms of what routing information is exchanged, which nodes exchange routing information, when routing information is updated, and how the routing information is transmitted. Finally, the traffic assignment process is the process of choosing where to send each data packet received.

## The Control Mechanism

Several types of control mechanisms may be used in a routing scheme: deterministic, isolated, centralized, and distributed. In deterministic routing, the routing information is assembled

---

(3) This method of describing and classifying routing algorithms is from John M. McQuillan, "Adaptive Routing Algorithms for Distributed Computer Networks," Bolt Beranek and Newman Inc., May 1974, pp. 170-184.

into the gateway software and does not change. This is the type of gateway routing currently implemented, but since we are concerned with designing a routing algorithm that is responsive to changes in the catenet, we did not consider it. Another type of routing control is isolated control, in which each gateway makes routing decisions based only on local information and no routing information is exchanged between the gateways. This type of control strategy was also not considered as changes in the catenet may not be detectable by isolated gateways.

In a centralized routing scheme, all the gateways send information about their local environment, e.g., connectivity or the state of their packet queues, to a central routing center. This routing center can be implemented either on one computer or on several, which are used to increase reliability. The routing center collects data from all gateways, determines the best route between all pairs of networks, and distributes routing information to the gateways to allow them to forward traffic on the chosen routes.

We have rejected use of a centralized control scheme for several reasons. There are inherent problems with centralized control schemes. For instance, if the routing center becomes inoperative, the network ceases to function properly; in particular, gateways cannot react to any further changes in the catenet. If several routing centers are used to increase reliability, the routing algorithm becomes more complex, because it must provide a method for coordinating use of the routing centers. In addition, the expense of developing and maintaining several routing centers may be prohibitive. Regardless of the number of routing centers implemented in a centralized scheme, there will be congestion around the routing centers and this congestion will become more severe as the number of gateways in the catenet increases. Finally, the coordination of the management of the routing center or centers is a problem. Unlike

the ARPANET where all the network components are under the control of one organization, the catenet consists of gateways and networks under the control of different organizations with different goals. These organizations may be unwilling to give control of the catenet to an organization or set of organizations that is responsible for the routing centers, and therefore, the performance of the catenet.

The proposed routing algorithm will use distributed control. In a distributed control scheme, each gateway exchanges routing information with other gateways in the catenet, and, using the information it has collected from other gateways, each gateway then determines its own best route to each destination. Thus, both the exchange of routing information and the routing calculation are distributed throughout the catenet.

A distributed control strategy has several advantages. First, it can detect and react to changes in the catenet, which makes it preferable to either a deterministic or isolated control scheme. Second, it distributes the exchange of routing information evenly throughout the catenet, thereby eliminating the problem of congestion around routing centers in a centralized scheme. Third, it distributes the computation required to determine routes through the catenet, thus eliminating the need for large and costly routing centers. Finally, it allows each organization to control and monitor its own gateways.

The Decision Process

The routing algorithm proposed is similar to a shortest path routing algorithm. In a shortest path routing algorithm, each node maintains an N x K table of the minimum costs to each of N destinations through each of K neighbors. These costs may be hop counts, delay, distance, reliability, etc. In the proposed algorithm, hop count is used. Using these minimum costs, each

node computes its own minimum cost to destination N as the minimum over K of its cost to neighbor K plus the cost from K to destination N. Packets for destination N are routed to the neighbor that is on the minimum cost path to destination N. To update the routing information, each node sends an N-entry vector of its minimum costs to all destinations to each of its K neighbors.

The original ARPANET routing strategy was a shortest path routing scheme as outlined above. Several problems were discovered in the ARPANET using this type of routing algorithm. In particular, the routing algorithm is slow to respond to changes in the network that increase the distance on the best path between two nodes. For example, consider the response of the routing algorithm in this three-node network when link B,C fails.

A-------B-------C

In this network, the minimum distance from A to C, d(A,C), is 2 hops and the minimum distance from B to C, d(B,C), is 1 hop. When link B,C fails, B will set its minimum distance to C to d(B,A) + d(A,C) = 3. Similarly, on receiving the new value for d(B,C), A will set d(A,C) = d(A,B) + d(B,C) = 4. A and B will increment their hop counts to C and continue to loop traffic for C through each other until the hop count to C is considered infinite. At this point, A and B will declare C unreachable.

A solution to this problem has been proposed.(4) Stated briefly, on receiving information that its best path to a destination has become worse, each node continues to report the hop count on this path for some amount of time, referred to as hold down time. In the network above, when link B,C fails, B reports its distance to C on path B,C as infinity rather than reporting the distance on a new path through A. The hold down time should be long enough for new routing information to propagate from the node detecting the

---

(4) McQuillan, op. cit., pp. 227-237.

change to nearby nodes and back. The distance in number of hops over which the routing information must be propagated and returned depends on the configuration of the network. When the hold down time expires, the node recomputes its minimum distances to all destinations and either finds the length of an alternate route or finds that the destination is unreachable.

Although the hold down mechanism solves the problem outlined above, further study of shortest path routing with hold down has shown that other problems arise.(5) Because of the uncertainty in the number of nodes that must receive new routing information and in the propagation time of routing messages, a node may leave hold down prematurely, before receiving new information from every node that may affect its selection of an alternate route. To solve this problem, we have developed an alternative to the hold down scheme. In this scheme, each node computes its minimum distance to all destinations, as in a shortest path algorithm. To update the routing information, each node sends an N-entry vector of distances to all destinations to each of its K neighbors. However, rather than reporting its minimum hop distance to each destination, each node composes K separate distance vectors and sends the Kth vector to the Kth neighbor. The entries in these distance vectors are computed as follows. If a node, G, has a minimum hop count to destination N that is less than or equal to the minimum hop count from its neighbor K to destination N, i.e., $d(G,N) \leq d(K,N)$, then the Nth entry of the distance vector is G's minimum hop count to destination N. If node G has a minimum hop count to destination N greater than the minimum hop count from neighbor K to destination N, then the Nth entry of the distance vector is infinity.

---

(5) John M. McQuillan, Gilbert Falk, and Ira Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm," Bolt Beranek and Newman Inc., pp. 25-26.

When connectivity in the catenet changes, a node can immediately use a route through a neighbor that was equidistant from the destination, i.e., d(G,N) = d(K,N). However, a node cannot immediately choose an alternate route through any other neighbor, as all neighbors not on a best route from the node to a destination have reported a distance of infinity. Instead, the node must wait until it has received new routing information from a set of nodes that can provide an alternate route. This mechanism removes the uncertainty of the hold down timer in estimating the time interval in which nodes should have reported new routing information. It also prevents nodes from looping packets indefinitely, as the distance from a node to any destination through itself will be infinity. The gateways will use this version of shortest path routing to determine the best route to each network. As packets are routed to networks rather than gateways, the gateways will maintain routing information on their shortest paths to networks rather than on paths to other gateways.

There are many possible choices for routing decision processes. This decision process was chosen because it can operate within the constraints of the catenet and satisfy the current goals for an internet routing strategy as outlined below.

1. The routing algorithm must be simple in order to reduce costs and to increase reliability. If the expense of interconnecting networks, either in purchasing gateway hardware or developing gateway software, becomes too great, fewer networks will be interconnected, and the catenet will be less functional. Special purpose gateways for connecting two specific networks may be developed, rather than the general purpose gateway described here, and the catenet will become even more fragmented. Also, since a gateway is the only link between networks, several gateways are often connected to each network to improve reliability. Such multiple connections are possible only if gateways are small and simple.

2. Gateways must provide high bandwidth connections between networks. Gateways may be used to connect two networks in which many large host computers communicate with hosts or terminals on other networks. As the gateways must support high throughput traffic, they must be able to forward traffic quickly and to respond quickly to changes in connectivity in the catenet, especially to changes that make a node unreachable via the path currently in use. If the traffic level between two networks is several hundred kilobits per second, a delay of one or two seconds in adapting the routing may be too great to prevent congestion and loss of a large number of packets. The proposed routing algorithm requires a minimum amount of processing in order to forward a packet and it can react quickly to failure.

3. As the gateways in the catenet will be controlled by several different organizations, some gateways may not use the adaptive routing algorithm proposed. It is assumed that gateways that do not implement the proposed algorithm will use a deterministic routing algorithm.(6) The proposed routing algorithm can then be extended to permit gateways that implement the algorithm to route traffic through any gateway. (The mechanism for permitting use of gateways that do not implement this algorithm is explained in the formal specification section below.)

4. It must be possible to add new gateways and new networks to the catenet dynamically without reassembling information in the existing gateways. As each gateway may be under the control of a different organization, it is unreasonable to require that all gateways be updated manually each time the configuration of the catenet changes. (The mechanism for adding new gateways under

---

(6) If two sets of gateways in the same catenet implement different adaptive routing schemes, no set of gateways can be guaranteed to deliver traffic properly. It is not possible to detect loops over several gateways formed from different responses by different routing algorithms to a change in the catenet.

the proposed routing algorithm is explained in the formal specification section below.)

5. It must be easy to modify the algorithm. The needs of the internetting community are changing. Currently, the primary goal in designing routing algorithms is to provide alternate routing around failed components. As the catenet expands, the need to provide mechanisms, such as minimum cost routing or access control, may become equally important. Also, a better understanding of routing problems, such as lockups and congestion, is leading to the design of improved routing algorithms. We will continue to study work in this field especially as it applies to solving the problems of routing in the catenet. Finally, experience in using this routing algorithm may point out aspects of the algorithm that should be modified. The proposed routing algorithm can be easily modified in several ways. For example, the decision process can be changed to a simple minimum cost algorithm, and the distance function can be changed to measure delay or cost. Similarly, the updating process can be altered independently of the decision process.

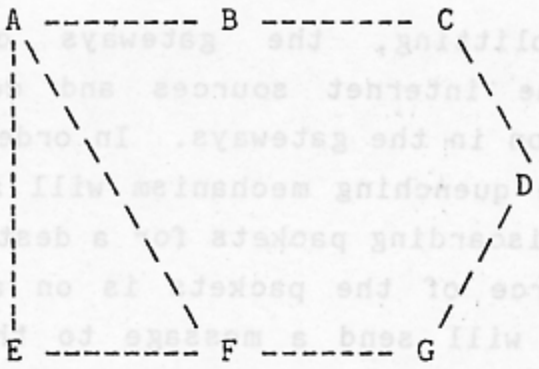
The Updating Process

The routing information in the gateways is updated as follows. Each gateway sends routing updates to each of its neighbors. The routing update is a vector of distances to each network, calculated as explained in the section on the decision process. The exchange of routing information is event-driven; events that cause routing updates are a change in connectivity between a gateway and its attached networks or neighbor gateways, the addition of a network or gateway to the catenet, and receipt of a routing update from a neighbor that changes the contents of the gateway's routing updates. Routing updates are exchanged reliably using a retransmission and acknowledgment scheme.

There are several reasons for using this type of updating process. Failures must be detected and reported quickly. As the gateways must support high throughput traffic, congestion can spread rapidly if traffic is sent on routes that no longer reach the destination. However, if routing updates are sent too often, either in response to a gateway or network cycling up or down or as the result of detecting a failure or recovery incorrectly, gateways may use a large percentage of the available bandwidth for exchanging routing updates. In order to react to failures quickly, the gateways will send routing updates whenever a failure is detected in a gateway or network. To avoid using excessive bandwidth for routing updates, a gateway will not use the information that another gateway or network has recovered until some time after it has recovered. Since the routing information deals only with connectivity changes and since this information is reported reliably, it will not be necessary to report routing information periodically, thus, the overhead of participating in adaptive routing is reduced.

The Traffic Assignment Process

The traffic assignment process will load split traffic on equal and minimum length paths to the destination network. The paths used to forward traffic must be of equal and minimum length to prevent looping as the following example illustrates.




Suppose that nodes A, E, and F are sending traffic to D. Node A is sending traffic on paths A,B,C,D, and A,F,G,D, node E is sending traffic on path E,F,G,D, and node F is sending traffic on path F,G,D. Node A detects that path A,B,C,D is congested and decides to send some traffic via path A,E,F,G,D. If node F detects that path F,G,D is congested and sends some traffic via A,B,C,D, then traffic will loop among nodes A,E, and F. This loop could be prevented by forcing all nodes to load split traffic only on minimum length paths to a destination.

Each gateway will maintain a list of neighbor gateways on equal and minimum length paths to each network. The gateway will forward packets for a given network through the set of neighbor gateways for that network, sending one packet to each neighbor in round-robin fashion. A more complex load splitting strategy could be implemented in order to utilize fully the gateways' capacities, to avoid congestion, and to provide the maximum throughput through the catenet. In such a strategy, the gateways could maintain averages of their input and output traffic rates and send information on these rates to their neighbor gateways. The gateways could use this information to split their traffic load in a way that would prevent congestion. We will continue to study such load splitting strategies and will recommend modifications to this algorithm to minimize congestion and increase capacity in the catenet.

Through load splitting, the gateways can provide better throughput to the internet sources and destinations and can decrease congestion in the gateways. In order to ease congestion further, a source quenching mechanism will also be implemented. If a gateway is discarding packets for a destination network, and the internet source of the packets is on an attached network, then the gateway will send a message to the source indicating that it should quench its flow for that destination network. If the source does not quench its flow, the first gateway to receive its packets will discard some of the packets to protect the catenet from additional congestion.

Comparison of Routing Algorithms

Over the past few months, several algorithms for routing packets through the catenet have been considered. In the following section, the proposed algorithm is compared to two algorithms presented in earlier papers.(7)(8)

One algorithm that was considered is a minimal delay routing algorithm developed by Robert Gallager.(9) This algorithm attempts to minimize the average delay for all packets in the catenet and to ensure that the route to each destination in the catenet is loop free. The algorithm is based on a shortest path routing scheme, using marginal delay between gateways as the cost function. In addition, a mechanism called "blocked status" is introduced to ensure loop free routing. Each node determines its blocked status with respect to every destination and sends this status in its routing updates. No node is permitted to send

---

(7) Virginia M. Strazisar, "Gateway Dynamic Routing," Bolt Beranek and Newman Inc., Jan. 1978, PRTN #241, PSPWN #98.

(8) Radia Perlman, "Gateway Routing," Bolt Beranek and Newman Inc., Jan. 1978, PRTN #242, PSPWN #99.

(9) Robert Gallager, "A Minimum Delay Routing Algorithm Using Distributed Computation," IEEE Transactions on Communication, Jan. 1977.

traffic to a destination through a node that is blocked with respect to that destination. The major disadvantages of this algorithm are the complexity of the algorithm and the difficulty of computing the marginal delay between gateways for use in the cost function. The proposed algorithm is easier to implement and the cost function, hop count, is easier to measure. Although the proposed algorithm will allow temporary loops in packet routes, it will not allow permanent loops to form. Gallager's algorithm potentially provides better routing, at least in minimizing delay, than the proposed algorithm. However, in an initial implementation, where the primary goal is to provide alternate routing, the cost of providing minimal delay routing using Gallager's algorithm is too high.

A routing algorithm based on the use of link-state information was also considered. This type of routing algorithm provides shortest path routing. In this routing algorithm, each gateway determines the connectivity between itself and its neighbors. To update the routing information, each gateway sends this connectivity information to each of its neighbors. Using the information obtained in the routing updates, each gateway maintains a matrix of the connectivity between all the gateways in the catenet. A gateway determines the shortest paths to each other gateway, by squaring the connectivity matrix, using the operations addition and minimum (e.g., matrix entry i,j is the minimum over k of entries i,k + k,j). A link-state algorithm and the algorithm proposed here both route traffic on the shortest path to the destination. However, a short comparison of the major differences between the proposed and link-state algorithms shows that the proposed algorithm requires less storage and computation in each gateway and requires the exchange of less routing information.

In the comparison below, the following symbols are used:

G = number of gateways in the catenet

N = number of networks in the catenet

K = average number of neighbor gateways

| | Link-State | Proposed |
|---|---|---|
| Storage | (G+N) x (G+N) connectivity matrix | N x K hop count matrix<br>K-entry vector of distance to neighbors<br>N-entry minimum hop count vector |
| Computation | G x G x G x log G operations to square matrix to compute shortest paths | N x K operations to compute minimum hop vector<br>N x K operations to compute routing updates |
| Updating | G x K packets containing connectivity to G gateways | G x K packets containing connectivity to N networks |

It is assumed that the number of gateways is greater than the number of networks and that the number of networks is much greater than the average number of neighbor gateways. Thus, from the table above, it can be seen that a link-state algorithm requires more storage, more computation and more updating information than the proposed algorithm, especially in the catenet, where the number of gateways should be several times the number of networks. Despite this drawback, a link-state algorithm does have advantages over the proposed routing algorithm. It may be more robust than the proposed routing algorithm. In a link-state algorithm, each gateway reports only its connectivity to other gateways and networks. If a gateway reports this connectivity incorrectly, packets may be incorrectly routed, and some packets may not be delivered to their destinations. However, the effect of one gateway reporting incorrect connectivity is not as great as in the proposed algorithm. In the proposed algorithm, a gateway could

incorrectly report that it is the minimum distance to all destinations. In this case, most packets would not be delivered to their destination. Although this is a drawback to the proposed algorithm, it is not a serious enough problem to warrant adopting a more costly link-state algorithm. If the proposed algorithm proves to be too unreliable, it can be improved by the addition of checksumming and consistency checking.

Formal Definition of the Basic Algorithm

The Decision Process

1. Each gateway contains a minimum distance matrix. The minimum distance matrix is a matrix with N x K entries, where N is the number of networks in the catenet and K is the number of neighbor gateways. Entry I,J, represented as d(I,J) is the number of hops to network I via neighbor J.

2. Each gateway computes a minimum distance vector of length N. This vector contains the minimum length in number of hops to each network from the gateway. The gateway calculates the Ith entry in its minimum distance vector, represented as d(I) as:

$$d(I) = \min \text{ over neighbors } J[d(G,J)+d(I,J)]$$

where d(G,J) is the distance from the gateway to the Jth neighbor, determined as in step 4. If the Ith network is physically attached to gateway G, then the value of the Ith entry is determined as in step 3.

3. Each gateway is responsible for determining its connectivity to its attached networks. If the gateway is able to receive and transmit traffic on attached network I, then it sets the Ith entry of its minimum distance vector to 0. If the gateway interface to I cannot send and receive traffic, then the gateway sets the Ith entry of its minimum distance vector to infinity. The method by which the gateway determines connectivity to its attached networks is dependent on the local implementation.

4. Each gateway is responsible for determining the connectivity to each of its neighbors. The distance from gateway G to neighbor J, d(G,J), is 1 if G is currently connected to J. If the connectivity between G and J is currently broken, then d(G,J) is infinity. A gateway determines connectivity to each neighbor by periodically sending internet packets to the neighbor with itself as the internet destination.

5. Each gateway, G, computes a routing table of neighbors through which to send packets for each network. The entry for network I contains all neighbors, J, such that the Ith entry of G's minimum distance vector equals the sum of the distance from G to neighbor J plus the distance from J to network I, i.e., $d(I) = d(G,J) + d(I,J)$. If G is physically attached to network I, then all entries in the routing table for I are zero; if network I is unreachable from G, then all entries in the routing table for I are infinity.

Updating Routing Information

6. Each gateway, G, computes routing update vectors. For each neighbor, J, construct a vector in which

entry I is the Ith entry of G's minimum path vector if

$d(G,J) \leq d(I,J)$

entry I is infinity if $d(G,J) > d(I,J)$

7. For each of K neighbors, send the Jth routing update vector to the Jth neighbor.

8. Each routing update is retransmitted until it is acknowledged. If the routing update is not acknowledged after a given number of retransmissions or within a given time period, the gateway assumes that connectivity to the neighbor to which it was sent is broken (see step 4). The retransmit and acknowledgement scheme is explained below.

9. The routing updates are sent when the contents of the routing update to any neighbor changes. A gateway sends an update when it detects that its connection to a network or to a neighbor gateway has failed; when it detects that connectivity to a network or neighbor has been reestablished for an amount of time; or when it receives a routing update from a neighbor that changes the contents of its routing updates.

10. On receipt of a routing update from neighbor J, the gateway overwrites the Jth row of the N x K minimum distance matrix with the contents of the routing update. The gateway recalculates its routing update vectors and sends these to its neighbors if the contents of any update have changed (see steps 6-9).

Traffic Assignment

11. Each gateway forwards a packet as follows. The gateway uses the network destination address from the packet's internet header as an index into the routing table (see step 5). The routing table contains a list of entries for each network; these entries should be used in round robin fashion, referencing the next entry for the next packet sent to the same network. There are three types of entries in the routing table:

1) The routing table entry is infinity. The network is unreachable, the packet is discarded, and the internet source of the packet is notified (see step 13).

2) The routing table entry is zero. The gateway is physically attached to the destination network. If the packet host address matches the gateway address on the destination network, then the packet is sent to the appropriate process in the gateway machine. If the host address does not match the gateway's, then the packet is sent on the gateway's interface to the destination network.

3) The routing table entry is a neighbor gateway. The gateway sends the packet to this neighbor.

12. Each internet source must decide to which gateway on its attached network to send packets for each other network. This routing decision may be made as follows. Each internet source maintains a list of gateways on its attached network. The source sends a packet destined for a network, I, to any gateway on its list. On receiving this packet, the gateway determines if any

gateway on the same network as the source is in its routing table entry for network I. If there is such a gateway, G, the gateway that received the packet forwards it to G and sends a message to the source indicating that it should send packets destined for network I to gateway G. The source should then send all packets for network I to gateway G. Note that if the routing tables in the gateways change so that gateway G is no longer on the best path to network I, the source will be given a new gateway to use for network I using the above procedure.

13. If a gateway determines that the destination network specified in a packet is unreachable, and the packet source and the gateway are on the same network, then the source cannot reach the destination. In this case, the gateway sends a message to the source indicating that the destination network is unreachable. On receipt of the message, the source should either stop sending traffic to the destination, readdress the traffic to the destination (if the destination has addresses on more than one network), or quench its traffic flow to the destination to avoid flooding the gateway with undeliverable traffic.

14. If a gateway receives a packet destined for network I and the gateway is dropping packets for network I, then the gateway reads the packet's internet source address. If the source and the gateway are on the same network, then the gateway sends a packet to the source indicating that it should quench its flow to network I.

Initialization

15. A gateway is initialized with the following information:

An address on an attached network for each of its K neighbors

An N x K minimum distance matrix with all entries set to infinity (see also step 17)

An N-entry minimum distance vector with all entries set to infinity

Its address on each network to which it is physically attached.

The gateway determines its connectivity to its attached networks and updates its minimum distance vector accordingly (see step 3). The gateway determines its connectivity to its neighbor gateways for use in calculating entries in the minimum distance vector (see step 4). Finally, the gateway computes routing updates and sends them as outlined above (see steps 6-9).

16. When a new gateway is added to the catenet, the neighbors of that gateway update their routing information as follows. When a gateway receives a routing update from a neighbor, it compares the internet source address in the packet to its list of its neighbors' addresses. If the packet source address does not match any of its neighbors' addresses, then it adds the packet source address to its list of neighbors and appends the routing update as a new row of the minimum distance matrix. The new gateway is then completely integrated into the catenet, as it is not necessary that any gateway that is not its neighbor know of its existence.

The following procedure is used to add a new network to the catenet. Each network is administratively assigned a network number. This network number is used as the index of the network in the minimum distance matrix, the minimum distance vector, the routing updates, etc. The length of these tables is determined by the highest assigned network number. Networks that are assigned numbers less than the highest network number, but that are not yet operational, are represented in all tables in the same manner as a network that is currently unreachable. If the length of a routing update received is greater than the length of its minimum distance matrix, then the gateway expands the minimum distance matrix to the length of the received routing update.

The gateway overwrites the appropriate row of its minimum distance matrix with the routing update. The gateway then sets all remaining new entries in the minimum distance matrix, formed by expanding the matrix, to infinity. Finally, the gateway recalculates its minimum distance vector and sends routing updates.

When a gateway is restarted, it contains a list of neighbors that existed when that gateway was assembled and a minimum distance matrix with a length equal to the highest network number assigned at the time the gateway was assembled. All other gateways and networks are new to this gateway and are added to its tables using the procedure outlined above.

Non-Routing Gateways

17. If a gateway is added to the catenet and that gateway does not participate in this routing scheme, the gateway can be used by gateways that do participate in routing as follows. For each non-routing gateway, make up an N-entry vector in which the Ith entry indicates whether or not the Ith network is reachable from this gateway. Manually assemble this vector into each neighbor of the new gateway that does participate in routing.

Gateways that participate in the proposed routing strategy compute their minimum distance vectors using only routing updates from other gateways that participate in the routing strategy. If an entry in its minimum distance vector is infinity, then a gateway recalculates that entry using the routing information it has for any non-routing neighbor gateways. Thus, non-routing gateways are used when they provide the only path to a destination.

1. Each gateway maintains a list of the most recent sequence numbers, R, that it has received in routing messages from each of its neighbors and a sequence number, S, for transmitting its routing messages.

2. On receiving a routing message from a neighbor, a gateway subtracts the most recent sequence number it has received from that neighbor, R, from the sequence number in the message, S.

A) If the value is greater than zero, the receiver accepts the data, acknowledges the sender's sequence number, S, and replaces its sequence number by S.

B) If the value is equal to zero, the receiver accepts the data and acknowledges its sequence number, R.

C) If the value is less than zero, the receiver rejects the data and acknowledges its sequence number, R.

On receiving an acknowledgement for a routing update, the gateway subtracts the sequence number acknowledged, R, from the sequence number, S, that it is using to transmit routing updates.

D) If the value is greater than zero, then an old routing update is being acknowledged. The receiver of the acknowledgement continues to retransmit the routing update.

E) If the value is zero, the receiver notes that the latest routing update has been accepted by the gateway that sent the acknowledgement.

F) If the value is less than zero, then the receiver of the acknowledgement replaces its sequence number with the sequence number in the packet. The receiver retransmits the routing update with the new sequence number.

3. Each time the routing information being sent by a gateway changes, the gateway increments its sequence number.

4. A gateway transmits its routing messages to all neighbors that have not acknowledged its current sequence number.

5. A gateway retransmits messages periodically until they are acknowledged. If they are not acknowledged after a given number of retransmissions or within a given time period, the neighbor to which they are being sent is declared unreachable. The number of retransmissions allowed or the time interval over which a message is retransmitted before a neighbor is declared unreachable is a function of the local implementation.

6. If a gateway is retransmitting a routing message and a new routing message is generated, the gateway stops retransmitting the old information and starts transmitting the new routing information. The sequence number used to transmit the routing information is incremented. The gateway resets any retransmission count or retransmission timeout value.

7. If the gateway's sequence number changes while it is retransmitting a message, it retransmits the message using the new sequence number. The gateway resets any retransmission count or retransmission timeout value.

8. The following procedure is used to initialize the transmission and receipt of routing messages. Initially, each gateway chooses a random sequence number, S, and assumes that the last sequence number acknowledged by any neighbor is not equal to S. The gateway uses sequence number S to transmit its first routing message to all its neighbors. The sequence number in the first routing message will either be greater, equal, or less than the last sequence number received from this gateway by a neighbor (corresponding to cases 2A through 2C above). If the sequence number is greater than a neighbor's sequence number, the neighbor accepts the data and acknowledges the sequence number in the packet. If the sequence number in the routing message is equal

to the last sequence number received by the neighbor, the neighbor accepts the routing message and acknowledges the sequence number in the packet. If the sequence number is less than the last sequence number received by the neighbor, the neighbor rejects the data and sends its sequence number in the acknowledgement. On receipt of this acknowledgement, the gateway sending the routing message advances to the neighbor's sequence number and retransmits the message using this sequence number. Thus, when all of its neighbors have received and acknowledged the first routing message, the gateway will have a sequence number that has been acknowledged by all its neighbors and that can be incremented and used by the gateway in future routing messages. On receipt of a routing message from a neighbor, a gateway that does not have a most recent sequence number received from that neighbor initializes the most recent sequence number received to the packet sequence number.

Debugging Aids

The protocol for sending gateway type messages also provides for two debugging aids. The first is a facility for tracing a packet's route through the catenet, and the second is a facility for specifying a packet's route at the source. Both of these facilities should be useful in debugging the adaptive routing scheme and in isolating failures in networks and gateways.

The trace facility functions as follows. Gateway type internet messages are marked as trace packets, and a trace pointer is initialized to an area of the packet where the trace information is to be stored. When a gateway receives a trace message, the gateway stores its address on the network from which it received the packet in the packet's trace area. The address is stored in the format specified for a source or destination address in the internet header. The trace pointer is altered to point past the new address. When the packet arrives at its destination, the

packet's trace area will include a complete copy of the packet's route through the gateways.

Source routing provides a method for the packet source to specify the packet's route. Gateway type internet messages are marked as source addressed, the source stores a route to the destination in the packet's data area, and initializes a route pointer to point to the first address. The entries in this route are the internet addresses of gateways along the route. On receiving a source routed packet, the gateway copies the next address in the route into the internet destination field and changes the route pointer to point to the next address. It then forwards the packet, using the normal forwarding algorithm. Note that the source need not specify all gateways along the route; in fact, the source cannot guarantee that the packet will traverse only the specified gateways, as the gateways will use the normal forwarding procedure to route packets between the source specified addresses.

Initially, both of these debugging aids will be used only in gateway type messages. If the debugging aids prove useful, they may later be defined as internet options.

Tables and Variables

Number of known networks, N

Number of known neighbors, K

A neighbor of a gateway is any gateway physically attached to the same network or networks to which this gateway is attached. If a gateway and a neighbor have M networks in common, then each gateway is considered to have M separate neighbors, one for each network that the gateways have in common.

Gateway addresses

Address of the gateway on each network to which it is physically attached.

Neighbor gateway addresses

An address for each neighbor gateway; this should be the address of the neighbor on a network to which the gateway is physically attached. If a gateway and its neighbor have more than one network in common, each gateway should have an address for its neighbor on each network the gateways have in common. Thus, in this case, a gateway will have more than one address for its neighbor; these addresses should be considered as separate neighbors.

Connectivity to neighbors

A K-entry vector; the Ith entry indicates whether or not the gateway is currently connected to its Ith neighbor.

Connectivity to networks

An N-entry vector; the Ith entry indicates whether or not the gateway is currently connected to the Ith network. A gateway is connected to a network only if it is physically attached to that network and its interface to that network is functioning.

Distance matrix

An N x K matrix of the routing updates received from the K neighbor gateways.

Minimum distance vector

An N-entry vector of minimum distances to each network, computed from the distance matrix, and the vectors of connectivity to neighbors and networks.

Routing updates

A set of K, N-entry vectors giving the distance from this gateway to each network. The Ith vector is sent to the Ith neighbor. This table is computed from the distance matrix, the minimum distance vector, and the vectors of connectivity to networks and neighbors.

Routing information from non-routing neighbor gateways

An N-entry vector for each neighbor gateway that does not participate in this routing strategy. The Ith entry of this vector indicates whether or not the Ith network is reachable through the neighbor gateway. These vectors are used only in gateways that have neighbors that do not participate in routing.

Routing table

A table containing, for each network, a list of the neighbor gateways on a minimum length path to that network. This table is constructed from the minimum distance matrix. If a table entry for network I is zero, then the gateway is physically attached to network I. If a table entry for network I is infinity, then network I is unreachable from the gateway.

Sequence number

The sequence number to use in transmitting routing updates from this gateway.

Sequence numbers for receiving updates

A list of sequence numbers received in the most recent routing updates from each of the neighbor gateways.

Events and Responses

1. Connectivity to a neighbor gateway or to an attached network changes.

    1. Update vector of connectivity to neighbors or networks.

    2. Update minimum distance vector.

    3. Recompute routing updates.

    4. If any routing update has changed, send routing updates to neighbors.

2. A routing update is received.

    1. Compare the source address of the routing update to the list of neighbor gateways; if this update is from a new neighbor, add the neighbor to the list of neighbor gateways and adjust all tables accordingly.

    2. Compare the number of known networks to the number of networks reported on in this routing update; if the number of networks in the routing update is greater than the number of known networks, adjust all tables to account for the new networks.

    3. Copy the routing update into the appropriate table of the minimum distance matrix.

    4. Recompute the minimum distance vector and the routing updates.

    5. If any routing update has changed, then send routing updates to all neighbors.

3. Retransmissions of a routing update to a neighbor are timed out. The connectivity to that neighbor has been broken, follow the steps in response to event 1 above.

4. An acknowledgment for a routing update is received.

   1. If the sequence number acknowledged does not equal the sequence number for transmitting updates, then retransmit the update.

   2. If the sequence number acknowledged equals the sequence number for transmitting updates, then note that the neighbor that sent the acknowledgement has received the latest routing update.

5. A data packet from a network is received.

   1. Forward the data packet using the information in the routing table.

   2. If the network is unreachable, send a network unreachable message to the internet source.

   3. If the gateway is dropping packets for the destination network, and the internet packet source is on a network to which the gateway is physically attached, send a source quench message to the internet source.

   4. If the internet source is on a network to which the gateway is physically attached, and the gateway is forwarding traffic through another gateway on the same network, send a message to the internet source to redirect traffic to that gateway.

List of Timeout Values

1. Minimum time that a network or gateway must be up before a routing update announcing its recovery is sent: 30 seconds.

2. The number of times to retransmit a routing update or the length of time over which to retransmit a routing update before a neighbor is declared to be disconnected: a parameter local to each gateway implementation.

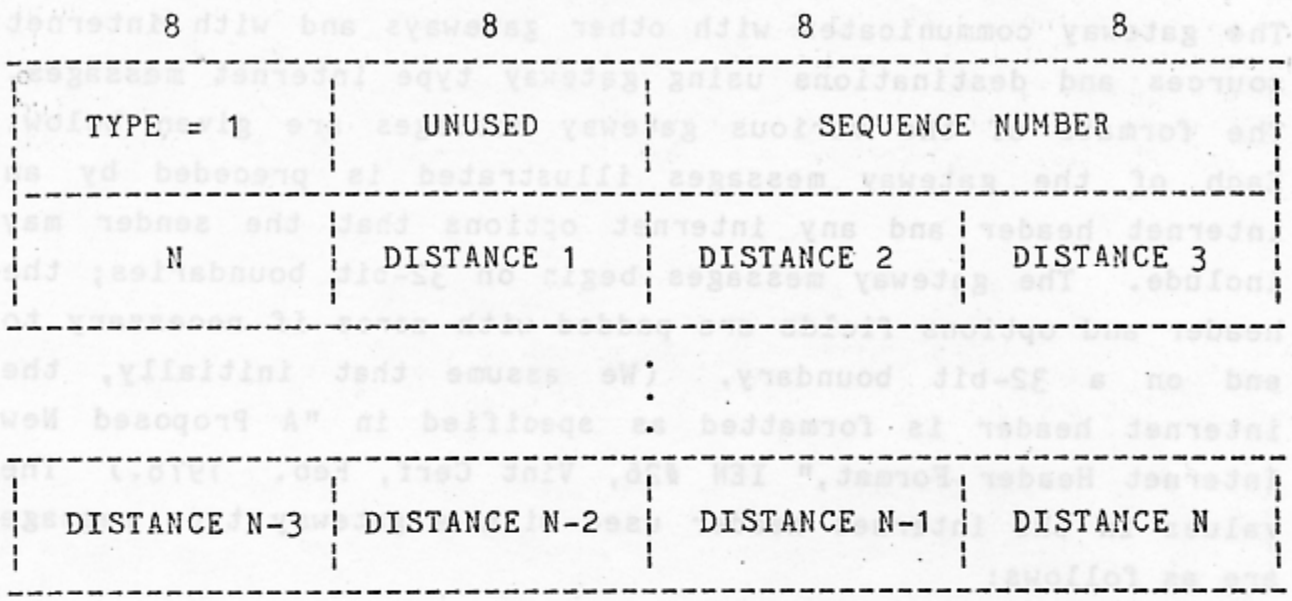
8. The minimum interval between sending destination network unreachable or source quench messages: a parameter local to each gateway implementation.

Message Formats

The gateway communicates with other gateways and with internet sources and destinations using gateway type internet messages. The formats of the various gateway messages are given below. Each of the gateway messages illustrated is preceded by an internet header and any internet options that the sender may include. The gateway messages begin on 32-bit boundaries; the header and options fields are padded with zeros if necessary to end on a 32-bit boundary. (We assume that initially, the internet header is formatted as specified in "A Proposed New Internet Header Format," IEN #26, Vint Cerf, Feb. 1978.) The values in the internet header used with a gateway type message are as follows:

| Field Name | Value |
|---|---|
| Version | 0 |
| Protocol | 3 |
| Option | Set if options are present. |
| Don't Fragment | Set in all gateway type messages. |
| TOS | Currently not used. |
| Packet Length | Length of internet header, options, and data in octets. |
| Internet Packet Id | 0 |
| MF | 0 |
| Fragment Number | 0 |
| Data Offset | 0 |
| Destination Net | Address of destination. |
| Destination Host | |
| Destination Port | In messages sent to gateways, this field is zero. In messages sent to a gateway to be echoed by that gateway, this field is set to 7. |
| Source Net | Address of source. |
| Source Host | |
| Source Port | |

ROUTING UPDATE




TYPE

The gateway message type. A routing update message is type 1.

SEQUENCE NUMBER

The 16-bit sequence number used to transmit routing updates.

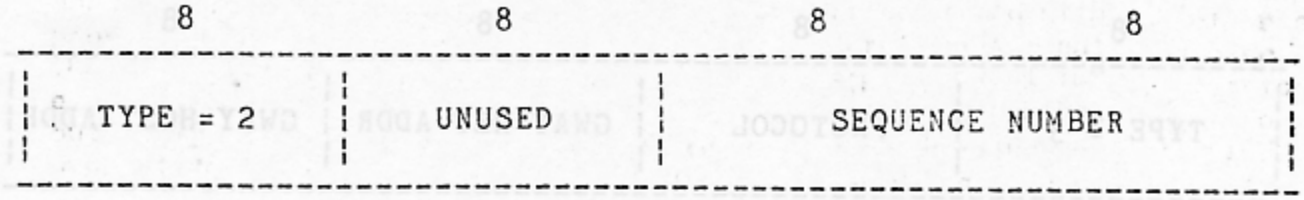
N

The number of networks reported on in this routing update.

DISTANCE 1...N

A set of 8-bit numbers that are the distances to each of the N networks. For example, if the ARPANET is assigned network number 10, then DISTANCE 10 is the distance to the ARPANET. (The currently assigned network numbers are listed in RFC #739, Jon Postel, Nov. 1977.) If the Ith network is unreachable, DISTANCE I is infinity, which is represented as octal 177.

ACKNOWLEDGEMENT



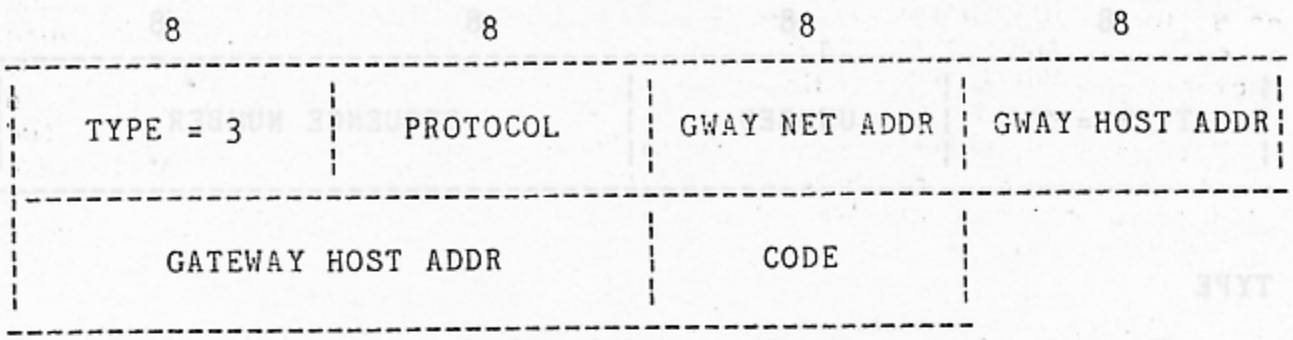

TYPE

The gateway message type. Acknowledgements are type 2.

SEQUENCE NUMBER

The 16-bit sequence number that the gateway is acknowledging.

DESTINATION UNREACHABLE




INTERNET HEADER

The internet header destination fields (net, host, and port) are copied from the internet header source fields of a message that the gateway is attempting to forward to the unreachable destination. The internet header source fields are copied from the internet header destination fields. Thus, this message will be sent to the source of a message that cannot be delivered, but will appear to be from that message's destination, rather than from the gateway. The internet protocol field is set to 3 as this is a gateway protocol message.

TYPE

The gateway message type. Destination unreachable messages are type 3.

PROTOCOL

This field is copied from the internet header protocol field of a message destined for the unreachable network or host.

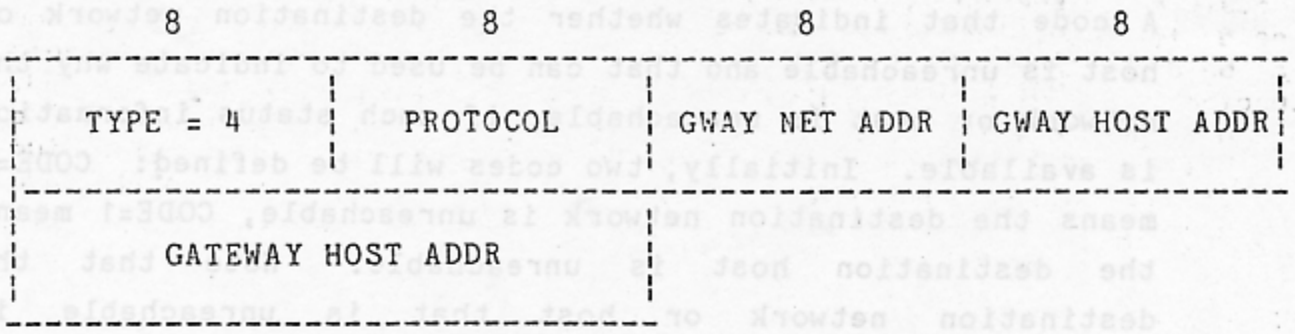
GWAY NET ADDR, GATEWAY HOST ADDR

The network and host address of the gateway sending the destination unreachable message. This should be the address of the gateway on the network on which the undeliverable message was received. This address is stored in the same format as an address in the internet header.

CODE

A code that indicates whether the destination network or host is unreachable and that can be used to indicate why the network or host is unreachable, if such status information is available. Initially, two codes will be defined: CODE=0 means the destination network is unreachable, CODE=1 means the destination host is unreachable. Note that the destination network or host that is unreachable is identified by the source address in the internet header.

SOURCE QUENCH




INTERNET HEADER

The internet header destination fields (net, host, and port) are copied from the internet header source fields of a message destined for a network for which the gateway is requesting that traffic be quenched. The internet header source fields are copied from the internet header destination fields. The internet protocol field is set to 3 as this is a gateway protocol message.

TYPE

The gateway message type. Source quench messages are type 4.

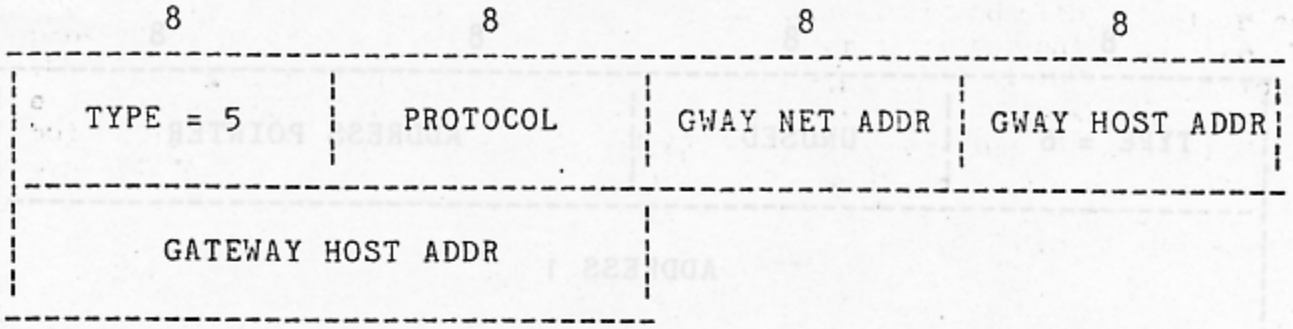
PROTOCOL

This field is copied from the internet header protocol field of a message destined for the network for which the gateway is attempting to quench traffic.

GWAY NET ADDR, GATEWAY HOST ADDR

The network and host address of the gateway sending the source quench message. This should be the address of the gateway on the network on which it received the traffic it is attempting to quench. This address is stored in the same format as an address in the internet header.

REDIRECT




INTERNET HEADER

The internet header destination fields (net, host, and port) are copied from the internet header source fields of a message destined for a network for which the gateway is attempting to redirect traffic. The internet header source fields are copied from the internet header destination fields. The internet protocol field is set to 3 as this is a gateway protocol message.

TYPE

The gateway message type. Redirect messages are type 5.

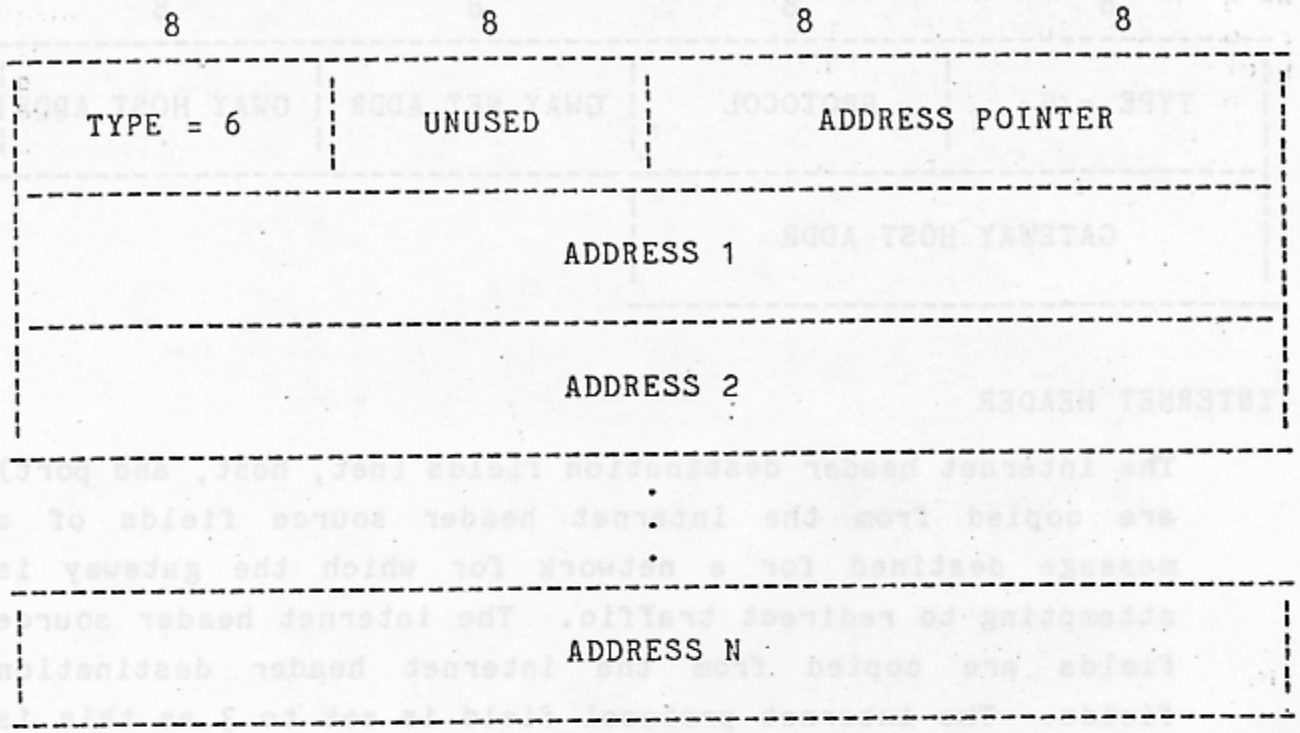
PROTOCOL

This field is copied from the internet header protocol field of a message destined for the network for which the gateway is attempting to redirect traffic.

GWAY NET ADDR, GATEWAY HOST ADDR

The network and host address of the gateway to which the traffic for the network specified in the internet header source net field should be sent. This address is stored in the same format as a network and host address in the internet header.

TRACE




TYPE

The gateway message type. Trace messages are type 6.

ADDRESS POINTER

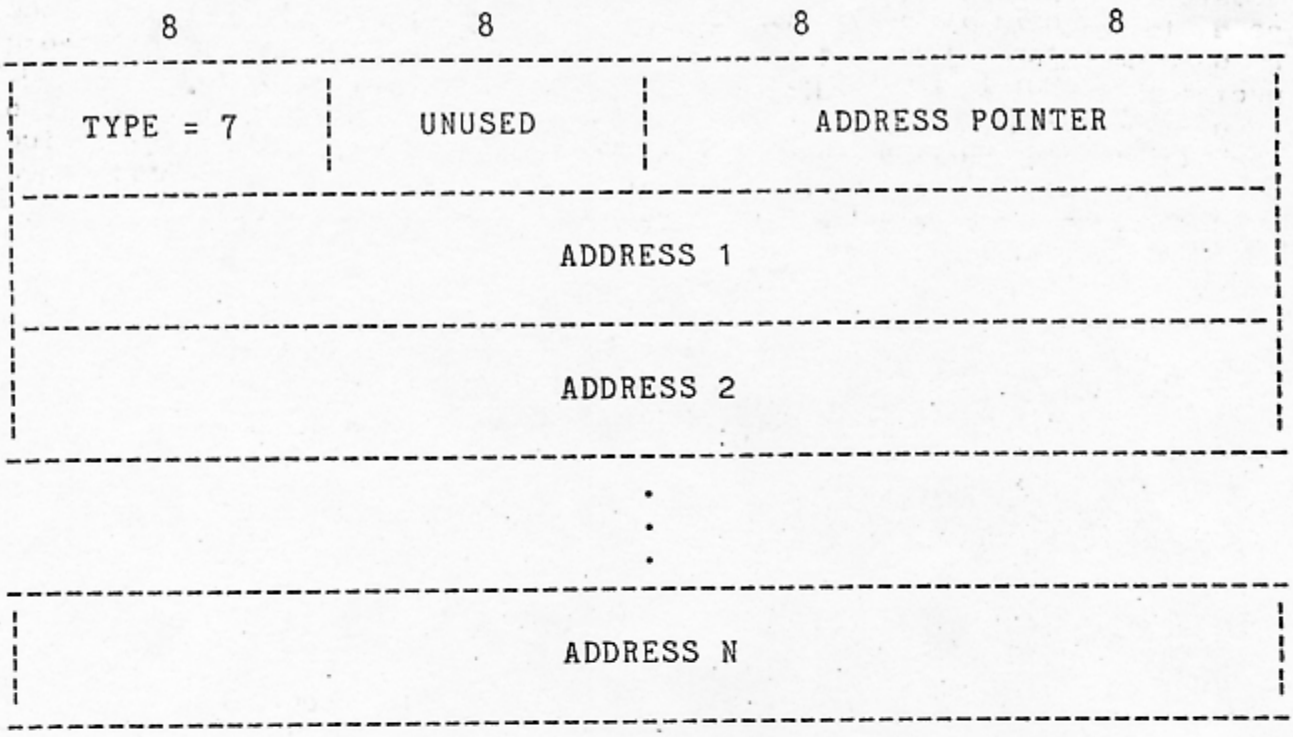
The address pointer is a 16-bit offset in bytes from the start of the packet to the place in the packet at which to store the next address. On receipt of a trace packet, a gateway stores its network and host address into the area of the packet referenced by the address pointer. The address pointer is incremented by the size of the address. If the address pointer points beyond the end of the packet, as indicated by the packet length field of the internet header, the gateway does not attempt to store its address in the packet.

ADDRESS 1...N

The address of the Ith gateway on the network from which it received the trace packet. The address is stored in the

same format as a network and host address in the internet header.

SOURCE ADDRESSED




TYPE

The gateway message type. Source addressed messages are type 7.

ADDRESS POINTER

The 16-bit address pointer is the offset from the start of the packet to the place in the packet from which to read the next address. On receipt of a source addressed packet, the gateway copies the address referenced by the address pointer into the internet header destination network and host address fields, and increments the address pointer field by the size of the address. The gateway sets the internet header destination port field to 0. If the address pointer points beyond the end of the packet, as indicated by the packet length field of the internet header, the gateway does not attempt to read the address in the packet.

ADDRESS 1...N

The address of the Ith gateway on the network on which it should receive the source addressed packet. The address is stored in the same format as a network and host address in the internet header.

Packets that are addressed to a gateway and that specify the destination port in the internet header as 7 will be echoed by the gateway. The gateway will swap the source and destination addresses in the internet header. These packets must include an internet header, but do not have to be gateway type messages.