Inter-Office Memorandum

To	Communication Protocols	Date	May 30, 1979
From	Ed Taft	Location	PARC/CSL
Subject	Gateway Information Protocol (revised)	File	[Maxc1]<Pup>GatewayInformation.press

XEROX

Attributes: technical, Distributed computing

Abstract: The Gateway Information Protocol is the means by which a host discovers the identity of its directly connected network(s) and locates gateways to other networks. This memo documents the protocol, presents algorithms for implementing it, and discusses some of its strengths and weaknesses.

The Gateway Information Protocol is the means by which a host discovers the identity of its directly connected network(s) and locates gateways to other networks. This topic is discussed at some length in another document [1]. This memo describes the actual protocol and algorithms presently employed for dealing with it.

The protocol

All gateway hosts implement a server on socket 2 that responds to requests for gateway information.

Gateway Information Request

Pup Type: 200 (octal)
Pup ID: arbitrary
Pup Contents: none

Gateway Information Response

Pup Type: 201 (octal)
Pup ID: same as in Request Pup
Pup Contents: one or more groups of four bytes, each providing routing information for one network, as follows:

<target-net> <gateway-net> <gateway-host> <hop-count>

In each group, the first byte specifies the target network number. If the gateway host is directly connected to that network, then the <hop-count> is zero and the <gateway-net> and <gateway-host> describe the gateway's connection to the network.

If the gateway host is not directly connected to the target network, then the second and third bytes give the network and host numbers of another gateway through which the responding gateway routes Pups to that network, and the fourth byte gives the hop count, i.e., the number of additional gateways (not including itself) through which the responding gateway believes a Pup must pass to reach the specified network. A hop count greater than the constant *maxHops* (presently 15) signifies that the target network is believed to be inaccessible.

It should be noted that the <gateway-net> and <gateway-host> bytes are not required by the routing table update algorithms. They are present for observation and debugging purposes.

Each gateway host must also periodically broadcast Gateway Information Pups, as described above, on all directly-connected networks. The frequency of this broadcast should be approximately one every 30 seconds, and immediately whenever the gateway's own routing table changes (see below). These Pups should be sent from socket 2 to socket 2.

Routing table maintenance

The following algorithms are presently used in all hosts, including gateways, for maintaining local routing tables. These algorithms serve two purposes:

1. To permit quick initialization of the local routing table when a host starts running Pup-based software.

2. To ensure that the local routing table remains up-to-date as inter-network topology changes (i.e., gateways come up or go down).

The local routing table contains an entry for each network to which it is believed to be possible to route Pups. (It is possible for a non-gateway host to maintain a subset of this routing table in the form of a cache, as discussed in a separate memo [2].) Each entry contains a hop count, the network and host numbers of a directly-connected gateway (if the hop count is nonzero), and a timeout used in the routing table maintenance algorithm. At initialization time, the routing table in a normal (non-gateway) host is empty; in a gateway host, it contains entries for all directly-connected networks (recall that gateway hosts *must* know the identity of all directly-connected networks).

When a Gateway Information Pup is received (either in response to the host's initial Gateway Information Request or gratuitously), the following procedure is used to process each four-byte entry in the Pup and update the local routing table. The network to which the entry refers (i.e., the one whose number is the first byte of the four-byte group) is referred to below as the *target network*.

First, a decision must be made whether or not it is appropriate to update the local routing table entry with the new information. If the target network is known to be directly connected (i.e., it exists in the local routing table with a hop count of zero), then no update is necessary. Otherwise, the update should be performed if any of the following conditions is true:

1. No entry for the target network presently exists in the local routing table.

2. The Gateway Information Pup's source address (network and host) is the same as the gateway address in the existing local routing table entry; that is, we are receiving updated information from the very gateway through which we are already routing Pups destined for the target network.

3. The existing entry in the local routing table has not been updated for some time, suggesting that it may no longer be valid. The timeout interval presently used is 90 seconds.

4. The hop count from the Gateway Information Pup, plus one, is less than the hop count in the existing local routing table entry; that is, the new information describes a means of reaching the target network through fewer intervening gateways than does the existing local routing table entry.

If any of these conditions holds, then the local routing table entry is updated (or a new one created) as follows:

1. The gateway network and host numbers in the entry are set to the source of the Gateway Information Pup.

2. The hop count in the entry is set to the Gateway Information Pup's hop count plus one. If this results in a hop count greater than *maxHops*, then the target network is deemed to have

become inaccessible.

3. The entry’s timeout is reset to 90 seconds.

In addition to processing Gateway Information Reply Pups, the routing table management software also periodically checks for routing table entries that have not been updated for so long that they are almost assuredly not valid. The timeout used for this is presently 3 minutes, or twice the 90-second interval after which the above algorithm deems an entry to be suspect. Entries not updated for 3 minutes are simply purged from the routing table.

Additional gateway procedures

When a gateway host’s routing table changes, the gateway should immediately broadcast a Gateway Information Pup reflecting the updated routing table, rather than waiting for the next periodic routing broadcast. This is so that changes in topology will propagate through the internet immediately rather than at a rate limited by the frequency of normal routing broadcasts.

The criteria for what constitutes a change’’ are important; incorrect choices may lead to oscillation or regenerative propagation of routing information. A gateway should immediately rebroadcast its routing table after updating it only in the following cases:

1. A formerly inaccessible network became accessible;

2. A formerly accessible network became inaccessible;

3. The hop count for some network changed.

In particular, a rebroadcast should *not* occur due to changing an entry to route through a different gateway with the same number of hops as before.

When a formerly accessible network becomes inaccessible, the gateway should continue to include the entry (with a hop count of *maxHops*+1, of course) in the routing tables it broadcasts during the next timeout interval (90 seconds). This ensures that knowledge of the network’s inaccessibility will propagate through the internet quickly.

When a gateway is about to go down voluntarily, it should broadcast several Gateway Information Pups in which all entries are marked inaccessible (i.e., hop count of *maxHops*+1), but then continue to forward Pups for a few seconds. This ensures that if alternate routes exist they will be found immediately rather than as a result of timeout.

Remarks on the protocol

The protocol and algorithms described above are adequate for maintaining routing information in an internetwork whose topology changes relatively infrequently. Because the routing table update algorithms are based on hop counts, they are stable while the internetwork topology is unchanging. And because changes are propagated immediately, the algorithms respond quickly when gateways come up or go down.

Routing changes caused by a gateway coming up are propagated immediately, i.e., at a rate not determined by timeouts but rather limited only by the store-and-forward propagation delay from the source of the change to each host along the shortest path. The change gives rise to a number of additional routing table broadcasts bounded by the number of single-hop paths between pairs of gateways in the internet.

When a gateway goes down voluntarily, the change in routing information also propagates immediately. In an ideal situation, where routing tables were passed around at a constant rate and synchronously, the change would propagate away from the gateway along the reverse of all routes directed through that gateway, invalidating such routes as it went. If another valid route to a network reached through that gateway existed, the change would stop propagating as soon as it reached a gateway knowing about that route, and the new route would then propagate back. This would give rise to a number of routing table broadcasts limited by twice the number of pairs of neighboring gateways.

In practice, routing information is passed around at variable rates and asynchronously. Consequently, a gateway may sometimes replace an invalid routing table entry with a route leading back through itself via some chain of other gateways that do not yet know the route is invalid. However, this is not a stable situation, since the bad routing entry's hop count is incremented as it is passed from gateway to gateway around the loop and quickly exceeds the maximum value of 15. Hence, the number of routing table broadcasts caused by a gateway going down is bounded by 15 times the number of pairs of neighboring gateways.

When a gateway goes down involuntarily, no routing information changes until some neighboring gateway times out a route directed through that gateway (i.e., after 90 seconds). At that time, the routing table entry becomes eligible for replacement by one from some other neighboring gateway. The first such routing table that is received (i.e., within the next 30 seconds) causes the routing table to change and thereby gives rise to immediate propagation of the change throughout the internet, as described previously.

The present routing table maintenance algorithm has the advantages of being relatively simple and imposing a very tiny computational burden on any single host (including a gateway). However, it has a number of defects that should be attacked in the next redesign:

It depends on all networks having a broadcast capability.

All known accessible networks are represented in every gateway's routing table. For a large internetwork, this would be impractical.

Every change in topology causes a flurry of activity throughout the internet. Again, for a large internet, this would be too costly.

The use of hop counts for determining optimal routes is appropriate only if store-and-forward delay is the principal limitation on performance. Where network bandwidth or gateway congestion is the major factor, one must actually measure the per-hop delay, as is done in the Arpanet. Note, though, that more sophisticated routing table update strategies (e.g, hold-down'') are required to ensure stability in this case. [3]

References

[1] E. Taft, Naming and Addressing Conventions for Pup,'' file [Maxc1]<Pup>PupName.press.

[2] E. Taft, Caching Pup Routing Information,'' file [Maxc1]<Pup>CacheRoute.press.

[3] J. McQuillan, *Adaptive Routine Algorithms for Distributed Computer Networks*, Harvard Ph.D. thesis, Report no. 2831, Bolt Beranek and Newman, May 1974.