Network Working Group
Request for Comments: 911

EGP GATEWAY UNDER BERKELEY UNIX 4.2

PAUL KIRTON

University of Southern California, Information Sciences Institute
Visiting Research Fellow from Telecom Australia Research Laboratories

22 August 1984

ABSTRACT

This report describes an implementation of the Exterior Gateway Protocol that runs under the Unix 4.2 BSD operating system. Some issues related to local network configurations are also discussed.

Status of this Memo:

This memo describes an implementation of the Exterior Gateway Protocol (EGP) (in that sense it is a status report). The memo also discusses some possible extentions and some design issues (in that sense it is an invitation for further discussion). Distribution of this memo is unlimited.

Funding for this research was provided by DARPA and Telecom Australia.

Table of Contents

1. INTRODUCTION

The Exterior Gateway Protocol (EGP) [Rosen 82; Seamonson & Rosen 84; Mills 84a] has been specified to allow autonomous development of different gateway systems while still maintaining global distribution of internet routing information. EGP provides a means for different autonomous gateway systems to exchange information about the networks that are reachable via them.

This report mainly describes an implementation of EGP that runs as a user process under the Berkeley Unix * 4.2 operating system run on a VAX ** computer. Some related issues concerning local autonomous system configurations are also discussed.

The EGP implementation is experimental and is not a part of Unix 4.2 BSD. It is anticipated that Berkeley will incorporate a version of EGP in the future.

The program is written in C. The EGP part is based on the C-Gateway code written by Liza Martin at MIT and the route management part is based on Unix 4.2 BSD route management daemon, "routed".

The EGP functions are consistent with the specification of [Mills 84a] except where noted.

A knowledge of EGP as described in [Seamonson & Rosen 84; Mills 84a] is assumed.

This chapter discusses the motivation for the project, Chapter 2 describes the gateway design, Chapter 3 is on testing, Chapter 4 suggests some enhancements and Chapter 5 discusses topology issues.

Further information about running the EGP program and describing the software is being published in an ISI Research Report ISI/RR-84-145 [Kirton 84].

Requests for documentation and copies of the EGP program should be sent to Joyce Reynolds (JKReynolds@USC-ISIF.ARPA). Software support is not provided.

1.1 Motivation for Development

With the introduction of EGP, the internet gateways will be divided into a "core" autonomous system (AS) of gateways maintained by Bolt, Beranek and Newman (BBN) and many "stub" AS's that are maintained by different organizations and have at least one network in common with a core AS gateway. The core AS will act as a hub for passing on routing information between

---

* Unix is a trade mark of AT&T

** VAX is a trade mark of Digital Equipment Corporation

different stub AS's so that it will only be necessary for stub AS's to conduct EGP with a core gateway. Further detail is given in [Rosen 82].

At the time of this project there were 28 "non-routing" gateways in the internet. Non-routing gateways did not exchange routing information but required static entries in the core gateway routing tables. Since August 1, 1984 these static entries have been eliminated and previously non-routing gateways are required to communicate this information to the core gateways dynamically via EGP [Postel 84].

At the USC Information Sciences Institute (ISI) there was a non-routing gateway to the University of California at Irvine network (UCI-ICS). With the elimination of non-routing gateways from the core gateway tables it is necessary to inform the core ISI gateway of the route to UCI-ICS using EGP.

Also, we would like a backup gateway between ISI-NET and the ARPANET in case the core ISI gateway is down. Such, a gateway would need to convey routing information via EGP. Details of the ISI network configuration are discussed in Section 5.2.

Of the 28 non-routing gateways 23 were implemented by Unix systems, including ISI's. Also, ISI's proposed backup gateway was a Unix system. Thus there was a local and general need for an EGP implementation to run under Unix. The current version of Unix that included Department of Defense (DoD) protocols was Berkeley Unix 4.2 so this was selected.

1.2 Overview of EGP

This report assumes a knowledge of EGP, however a brief overview is given here for completeness. For further details refer to [Rosen 82] for the background to EGP, [Seamonson & Rosen 84] for an informal description, and [Mills 84a] for a more formal specification and implementation details.

EGP is generally conducted between gateways in different AS's that share a common network, that is, neighbor gateways.

EGP consists of three procedures, neighbor acquisition, neighbor reachability and network reachability.

Neighbor acquisition is a two way handshake in which gateways agree to conduct EGP by exchanging Request and Confirm messages which include the minimum Hello and Poll intervals. Acquisition is terminated by exchanging Cease and Cease-ack messages.

Neighbor reachability is a periodic exchange of Hello commands and I-H-U (I heard you) responses to ensure that each gateway is up. Currently a 30 second minimum interval is used across ARPANET. Only one gateway need send commands as the other can use them to determine reachability. A gateway sending reachability commands is said to be in the active mode, while a gateway that just responds is in the passive mode.

Network reachability is determined by periodically sending Poll commands and receiving Update responses which indicate the networks reachable via one or more gateways on the shared network. Currently 2 minute minimum interval is used across ARPANET.

2. GATEWAY DESIGN

EGP is a polling protocol with loose timing constraints. Thus the only gateway function requiring good performance is packet forwarding. Unix 4.2 already has packet forwarding built into the kernel where best performance can be achieved. At the time of writing Unix 4.2 did not send ICMP (Internet Control Message Protocol) redirect messages for misrouted packets. This is a requirement of internet gateways and will later be added by Berkeley.

The EGP and route update functions are implemented as a user process. This facilitates development and distribution as only minor changes need to be made to the Unix kernel. This is a similar approach to the Unix route distribution program "routed" [Berkeley 83] which is based on the Xerox NS Routing Information Protocol [Xerox 81].

2.1 Routing Tables

A route consists of a destination network number, the address of the next gateway to use on a directly connected network, and a metric giving the distance in gateway hops to the destination network.

There are two sets of routing tables, the kernel tables (used for packet forwarding) and the EGP process tables. The kernel has separate tables for host and network destinations. The EGP process only maintains the network routing tables. The EGP tables are updated when EGP Update messages are received. When a route is changed the kernel network tables are updated via the SIOCADDRT and SIOCDELRT ioctl system calls. At initialization the kernel network routing tables are read via the kernel memory image file, /dev/kmem, and copied into the EGP tables for consistency.

This EGP implementation is designed to run on a gateway that is also a host. Because of the relatively slow polling to obtain route updates it is possible that the host may receive notification of routing changes via ICMP redirects before the EGP process is notified via EGP. Redirects update the kernel tables directly. The EGP process listens for redirect messages on a raw socket and updates its routing tables to keep them consistent with the kernel.

The EGP process routing tables are maintained as two separate tables, one for exterior routes (via different AS gateways) and one for interior routes (via the gateways of this AS). The exterior routing table is updated by EGP Update messages. The interior routing table is currently static and is set at initialization time. It includes all directly attached nets, determined by the SIOCGIFCONF ioctl system call and any interior non-routing gateways read from the EGP initialization file, EGPINITFILE. The interior routing table could in future be updated dynamically by an Interior Gateway Protocol (IGP).

Maintaining separate tables for exterior and interior routing facilitates the preparation of outgoing Update messages which only contain interior routing information [Mills 84b]. It also permits alternative external routes to the internal routes to be saved as a backup in case an interior route fails. Alternate routes are flagged, RTS_NOTINSTALL, to indicate that the kernel

routes should not be updated. In the current implementation alternate routes are not used.

2.1.1 Incoming Updates

EGP Updates are used to update the exterior routing table if one of the following is satisfied:

- No routing table entry exists for the destination network and the metric indicates the route is reachable (< 255).
- The advised gateway is the same as the current route.
- The advised distance metric is less than the current metric.
- The current route is older (plus a margin) than the maximum poll interval for all acquired EGP neighbors. That is, the route was omitted from the last Update.

If any exterior route entry, except the default route, is not updated by EGP within 4 minutes or 3 times the maximum poll interval, whichever is the greater, it is deleted.

If there is more than one acquired EGP neighbor, the Update messages received from each are treated the same way in the order they are received.

In the worst case, when a route is changed to a longer route and the old route is not first notified as unreachable, it could take two poll intervals to update a route. With the current poll interval this could be 4 minutes. Under Unix 4.2 BSD TCP connections (Transmission Control Protocol) are closed automatically after they are idle for 6 minutes. So this worst case will not result in the automatic closure of TCP connections.

2.1.2 Outgoing Updates

Outgoing Updates include the direct and static networks from the interior routing table, except for the network shared with the EGP neighbor.

The networks that are allowed to be advised in Updates may be specified at initialization in EGPINITFILE. This allows particular routes to be excluded from exterior updates in cases where routing loops could be a problem. Another case where this option is necessary, is when there is a non-routing gateway belonging to a different AS which has not implemented EGP yet. Its routes may need to be included in the kernel routing table but they are not allowed to be advised in outgoing updates.

If the interior routing table includes other interior gateways on the network shared with the EGP neighbor they are include in Updates as the appropriate

first hop to their attached networks.

The distance to networks is set as in the interior routing table except if the route is marked down in which case the distance is set to 255. At present routes are only marked down if the outgoing interface is down. The state of all interfaces is checked prior to preparing each outgoing Update using the SIOCGIFFLAGS ioctl system call.

Unsolicited Updates are not sent.

2.2 Neighbor Acquisition

EGPINITFILE lists the addresses of trusted EGP neighbor gateways, which are read at initialization. These will usually be core gateways as only core gateways provide full internet routing information. At the time of writing there were three core gateways on ARPANET which support EGP, CSS-GATEWAY, ISI-GATEWAY and PURDUE-CS-GW, and two on MILNET, BBN-MINET-A-GW and AERONET-GW.

EGPINITFILE also includes the maximum number of these gateways that should be acquired at any one time. This is usually expected to be just one. If this gateway is declared down another gateway on the list will then be acquired automatically in sufficient time to ensure that the current routes are not timed out.

The gateway will only accept acquisitions from neighbors on the trusted list and will not accept them if it already has acquired its maximum quota. This prevents Updates being accepted from possibly unreliable sources.

The ability to acquire core gateways that are not on the trusted list but have been learned of indirectly via Update messages is not included because not all core gateways run EGP.

New acquisition Requests are sent to neighbors in the order they appear in EGPINITFILE. No more new Requests than the maximum number of neighbors yet to be acquired are sent at once. Any number of outstanding Requests are retransmitted at 32 second intervals up to 5 retransmissions each at which time the acquisition retransmission interval is increased to 4 minutes. Once the maximum number of neighbors has been acquired, unacquired neighbors with outstanding Requests are sent Ceases. This approach provides a compromise between fast response when neighbors do not initially respond and a desire to minimize the chance that a neighbor may be Ceased after it has sent a Confirm but before it has been received. If the specified maximum number of neighbors cannot be acquired, Requests are retransmitted indefinitely to all unacquired neighbors.

2.3 Hello and Poll Intervals

The Request and Confirm messages include minimum values for Hello and Poll intervals. The advised minimums by this and the core gateways are currently 30 and 120 seconds respectively.

The received intervals are checked against upper bounds to guard against nonsense values. The upper bounds are currently set at 120 and 480 seconds respectively. If, they are exceeded the particular neighbor is considered bad and not sent further Requests for one hour. This allows the situation to be corrected at the other gateway and normal operation to automatically resume from this gateway without an excess of unnecessary network traffic.

The actual Hello and Poll intervals are chosen by first selecting the maximum of the intervals advised by this gateway and its peer. A 2 second margin is then added to the Hello interval to take account of possible network delay variations and the Poll interval is increased to the next integer ratio of the Hello interval. This results in 32 second Hello and 128 second Poll intervals.

If an Update is not received in response to a Poll, at most one repoll (same sequence number) is sent instead of the next scheduled Hello.

2.4 Neighbor Cease

If the EGP process is sent a SIGTERM signal via the Kill command, all acquired neighbors are sent Cease(going down) commands. Ceases are retransmitted at the hello interval at most 3 times. Once all have either responded with Cease-acks or been sent three retransmitted Ceases the process is terminated.

2.5 Neighbor Reachability

Only active reachability determination is implemented. It is done as recommended in [Mills 84a] with a minor variation noted below.

A shift register of responses is maintained. For each Poll or Hello command sent a zero is shifted into the shift register. If a response (I-H-U, Update or Error) is received with the correct sequence number the zero is replaced by a one. Before each new command is sent the reachability is determined by examining the last four entries of the shift register. If the neighbor is reachable and <= 1 response was received the neighbor is considered unreachable. If the neighbor is considered unreachable and >= 3 responses were received it is now considered reachable.

A neighbor is considered reachable immediately after acquisition so that the first poll received from a core gateway (once it considers this gateway reachable) will be responded to with an Update. Polls are not sent unless a neighbor is considered reachable and it has not advised that it considers this gateway unreachable in its last Hello, I-H-U or Poll message. This prevents the first Poll being discarded after a down/up transition. This is important as the Polls are used for reachability determination. Following acquisition at least one message must be received before the first Poll is sent. This is to determine that the peer does not consider this gateway down. This usually requires at least one Hello to be sent prior to the first poll. The discussion of this paragraph differs from [Mills 84a] which recommends that a peer be considered down following acquisition and Polls may be sent as soon as the peer is considered up. This is the only significant departure from the

recommendations in [Mills 84a].

Polls received by peers that are considered unreachable are sent an Error response which allows their reachability determination to progress correctly. This action is an option within [Mills 84a].

When a neighbor becomes unreachable all routes using it as a gateway are deleted from the routing table. If there are known unacquired neighbors the unreachable gateway is ceased and an attempt is made to acquire a new neighbor. If all known neighbors are acquired the reachability determination is continued for 30 minutes ([Mills 84a] suggests 60 minutes) after which time the unreachable neighbor is ceased and reacquisition attempted every 4 minutes. This is aimed at reducing unnecessary network traffic.

If valid Update responses are not received for three successive polls the neighbor is ceased and an alternative acquired or reacquisition is attempted in 4 minutes. This provision is provided in case erroneous Update data formats are being sent by the neighbor. This situation did occur on one occasion during testing.

2.6 Sequence Numbers

Sequence numbers are managed as recommended in [Mills 84a]. Single send and receive sequence numbers are maintained for each neighbor. The send sequence number is initialized to zero and is incremented before each new Poll (not repoll) is sent and at no other time. The send sequence number is used in all commands. The receive sequence number is maintained by copying the sequence number of the last Request, Hello, or Poll command received from a neighbor. This sequence number is used in outgoing Updates. All responses (including Error responses) return the sequence number of the message just received.

2.7 Treatment of Excess Commands

If more than 20 commands are received from a neighbor in any 8 minute period the neighbor is considered bad, Ceased and reacquisition prevented for one hour.

At most one repoll (same sequence number) received before the poll interval has expired (less a 4 second margin for network delay variability) is responded to with an Update, others are sent an Error response. When an Update is sent in response to a repoll the unsolicited bit is not set, which differs from the recommendation in [Mills 84a].

2.8 Inappropriate Messages

If a Confirm, Hello, I-H-U, Poll or Update is received from any gateway (known or unknown) that is in the unacquired state, synchronization has probably been lost for some reason. A Cease(protocol violation) message is sent to try and reduce unnecessary network traffic. This action is an option in [Mills 84a].

2.9 Default Gateway

A default gateway may be specified in EGPINITFILE. The default route (net 0 in Unix 4.2 BSD) is used by the kernel packet forwarder if there is no specific route for the destination network. This provides a final level of backup if all known EGP neighbors are unreachable. This is especially useful if there is only one available EGP neighbor, as in the ISI case, Section 5.2.2.

The default route is installed at initialization and deleted after a valid EGP Update message is received. It is reinstalled if all known neighbors are acquired but none are reachable, if routes time out while there are no EGP neighbors that are acquired and reachable, and prior to process termination.

It is deleted after a valid EGP Update message is received because the default gateway will not know any more routing information than learned via EGP. If it were not deleted, all traffic to unreachable nets would be sent to the default gateway under Unix 4.2 forwarding strategy.

The default gateway should normally be set to a full-routing core gateway other than the known EGP neighbor gateways to give another backup in case all of the EGP gateways are down simultaneously.

3. TESTING

A few interesting cases that occurred during testing are briefly described.

The use of sequence numbers was interpreted differently by different implementers. Consequently some implementations rejected messages as having incorrect sequence numbers, resulting in the peer gateway being declared down. The main problem was that the specification was solely in narrative form which is prone to inconsistencies, ambiguities and incompleteness. The more formal specification of [Mills 84a] has eliminated these ambiguities.

When testing the response to packets addressed to a neighbor gateway's interface that was not on the shared net a loop resulted as both gateways repeatedly exchanged error messages indicating an invalid interface. The problem was that both gateways were sending Error responses after checking the addresses but before the EGP message type was checked. This was rectified by not sending an Error response unless it was certain that the message was not itself an Error response.

On one occasion a core gateway had some form of data error in the Update messages which caused them to be rejected even though reachability was being satisfactorily conducted. This resulted in all routes being timed out. The solution was to count the number of successive Polls that do not result in valid Updates being received and if this number reaches 3 to Cease EGP and attempt to acquire an alternative gateway.

Another interesting idiosyncrasy, reported by Mike Karels at Berkeley, results from having multiple gateways between MILNET and ARPANET. Each ARPANET host has an assigned gateway to use for access to MILNET. In cases where the EGP gateway is a host as well as a gateway, the EGP Update messages may indicate a different MILNET/ARPANET gateway from the assigned one. When the host/gateway originates a packet that is routed via the EGP reported gateway, it will receive a redirect to its assigned gateway. Thus the MILNET gateway can keep being switched between the gateway reported by EGP and the assigned gateway. A similar thing occurs when using routes to other nets reached via MILNET/ARPANET gateways.

4. FUTURE ENHANCEMENTS

4.1 Multiple Autonomous Systems

The present method of acquiring a maximum number of EGP neighbors from a trusted list implies that all the neighbors are in the same AS. The intention is that they all be members of the core AS. When updating the routing tables, Updates are treated independently with no distinction made as to whether the advised routes are internal or external to the peer's AS. Also, routing metrics are compared without reference to the AS of the source.

If EGP is to be conducted with additional AS's beside the core AS, all neighbors on the list would need to be acquired in order to ensure that gateways from both AS's were always acquired. This results in an unnecessary excess of EGP traffic if redundant neighbors are acquired for reliability. A more desirable approach would be to have separate lists of trusted EGP gateways and the maximum number to be acquire, for each AS. Routing entries would need to have the source AS added so that preference could be given to information received from the owning AS (see Section 5.1.2)

4.2 Interface Monitoring

At present, interface status is only checked immediately prior to the sending of an Update in response to a Poll. The interface status could be monitored more regularly and an unsolicited Update sent when a change is detected. This is one area where the slow response of EGP polling could be improved. This is of particular interest to networks that may be connected by dial-in lines. When such a network dials in, its associated interface will be marked as up but it will not be able to receive packets until the change has been propagated by EGP. This is one case where the unsolicited Update message would help, but there is still the delay for other non-core gateways to poll core EGP gateways for the new routing information.

This was one case where it was initially thought that a kernel EGP implementation might help. But the kernel does not presently pass interface status changes by interrupts so a new facility would need to be incorporated. If this was done it may be just as easy to provide a user level signal when an interface status changes.

4.3 Network Level Status Information

At present, network level status reports, such as IMP Destination Unreachable messages, are not used to detect changes in the reachability of EGP neighbors or other neighbor gateways. This information should be used to improve the response time to changes.

4.4 Interior Gateway Protocol Interface

At present any routing information that is interior to the AS is static and read from the initialization file. The internal route management functions have been written so that it should be reasonably easy to interface an IGP for dynamic interior route updates. This is facilitated by the separation of the exterior and interior routing tables.

The outgoing EGP Updates will be correctly prepared from the interior routing table by rt_NRnets() whether or not static or dynamic interior routing is done. Functions are also provided for looking up, adding, changing and deleting internal routes, i.e. rt_int_lookup(), rt_add(), rt_change() and rt_delete() respectively.

The interaction of an IGP with the current data structures basically involves three functions: updating the interior routing table using a function similar to rt_NRupdate(), preparing outgoing interior updates similarly to rt_NRnets(), and timing out interior routes similarly to rt_time().

5. TOPOLOGY ISSUES

5.1 Topology Restrictions and Routing Loops

5.1.1 Background

EGP is not a routing algorithm. it merely enables exterior neighbors to exchange routing information which is likely to to be needed by a routing algorithm. It does not pass sufficient information to prevent routing loops if cycles exist in the topology [Rosen 82].

Routing loops can occur when two gateways think there are alternate routes to reach a third gateway via each other. When the third gateway goes down they end up pointing to each other forming a routing loop. Within the present core system, loops are broken by counting to "infinity" (the internet diameter in gateway hops). This (usually) works satisfactorily because GGP propagates changes fairly quickly as routing updates are sent as soon as changes occur. Also the diameter of the internet is quite small (5) and a universal distance metric, hop count, is used. But this will be changed in the future.

With EGP, changes are propagated slowly. Although a single unsolicited NR message can be sent, it won't necessarily be passed straight on to other gateways who must hear about it indirectly. Also, the distance metrics of different AS's are quite independent and hence can't be used to count to infinity.

The initial proposal was to prevent routing loops by restricting the topology of AS's to a tree structure so that there are no multiple routes via alternate AS's. Multiple routes within the same AS are allowed as it is the interior routing strategies responsibility to control loops.

[Mills 84b] has noted that even with the tree topology restriction, "we must assume that transient loops may form within the core system from time to time and that this information may escape to other systems; however, it would be expected that these loops would not persist for very long and would be broken in a short time within the core system itself. Thus a loop between non-core systems can persist until the first round of Update messages sent to the other systems after all traces of the loop have been purged from the core system or until the reachability information ages out of the tables, whichever occurs first".

With the initial simple stub EGP systems the tree topology restriction could be satisfied. But for the long term this does not provide sufficient robustness.

[Mills 83] proposed a procedure by which the AS's can dynamically reconfigure themselves such that the topology restriction is always met, without the need for a single "core" AS. One AS would own a shared net and its neighbor AS's would just conduct EGP with the owner. The owner would pass on such information indirectly as the core system does now. If the owning AS is defined to be closest to the root of the tree topology, any haphazard interconnection can

form itself into an appropriate tree structured routing topology. By routing topology I mean the topology as advised in routing updates. There may well be other physical connections but if they are not advised they will not be used for routing. Each AS can conduct EGP with at most one AS that owns one of its shared nets. Any AS that is not conducting EGP over any net owned by another AS is the root of a subtree. It may conduct EGP with just one other AS that owns one of its shared nets. This "attachment" combines the two subtrees into a single subtree such that the overall topology is still a tree. Topology violations can be determined because two different AS's will report that they can reach the same net.

With such a dynamic tree, there may be preferred and backup links. In such cases it is necessary to monitor the failed link so that routing can be changed back to the preferred link when service is restored.

Another aspect to consider is the possibility of detecting routing loops and then breaking them. Expiration of the packet time-to-live (TTL) could be used to do this. If such a loop is suspected a diagnostic packet, such as ICMP echo, could be sent over the suspect route to confirm whether it is a loop. If a loop is detected a special routing packet could be sent over the route that instructs each gateway to delete the route after forwarding the packet on. The acceptance of new routing information may need to be delayed for a hold down period. This approach would require sensible selection of the initial TTL. But this is not done by many hosts.

5.1.2 Current Policy

Considering the general trend to increased network interconnection and the availability of alternative long-haul networks such as ARPANET, WBNET (wideband satellite network), and public data networks the tree topology restriction is generally unacceptable. A less restrictive topology is currently recommended. The following is taken from [Mills 84b].

EGP topological model:

- An autonomous system consists of a set of gateways connected by networks. Each gateway in the system must be reachable from every other gateway in its system by paths including only gateways in that system.

- A gateway in a system may run EGP with a gateway in any other system as long as the path over which EGP itself is run does not include a gateway in a third system.

- The "core system" is distinguished from the others by the fact that only it is allowed to distribute reachability information about systems other than itself.

- At least one gateway in every system must have a net in common with a gateway in the core system.

- There are no topological or connectivity restrictions other than those implied above.

A gateway will use information derived from its configuration (directly connected nets), the IGP of its system, called S in the following, (interior nets) and EGP (interior and exterior nets of neighboring systems) to construct its routing tables. If conflicts with respect to a particular net N occur, they will be resolved as follows:

- If N is directly connected to the gateway, all IGP and EGP reports about N are disregarded.

- If N is reported by IGP as interior to S and by EGP as either interior or exterior to another system, the IGP report takes precedence.

- If N is reported by EGP as interior to one system and exterior to another, the interior report takes precedence.

- If N is reported as interior by two or more gateways of the same system using EGP, the reports specifying the smallest hop count take precedence.

- In all other cases the latest received report takes precedence.

Old information will be aged from the tables.

The interim model provides an acceptable degree of self-organization. Transient routing loops can occur between systems, but these are eventually broken by old reachability information being aged out of the tables. Given the fact that transient loops can occur due to temporary core-system loops, the additional loops that might occur in the case of local nets homed to multiple systems does not seem to increase the risk significantly.

5.2 Present ISI Configuration

A simplified version of the ISI network configuration is shown in Figure 5-1. ISI-Hobgoblin can provide a backup gateway function to the core ISI-Gateway between ARPANET and ISI-NET. ISI-Hobgoblin is a VAX 11/750 which runs Berkeley Unix 4.2. The EGP implementation described in this report is run on ISI-Hobgoblin.

ISI-Troll is part of a split gateway to the University of California at Irvine network (UCI-ICS). The complete logical gateway consists of ISI-Troll, the 9600 baud link and UCI-750A [Rose 84]. ISI-Troll runs Berkeley Unix 4.1a and hence cannot run the EGP program. It is therefore a non-routing gateway. The existence of UCI-ICS net must be advised to the core AS by ISI-Hobgoblin. This can be done by including an appropriate entry in the EGPINITFILE.

Hosts on ISI-NET, including ISI-Troll, have static route entries indicating ISI-Gateway as the first hop for all networks other than UCI-ICS and ISI-NET.

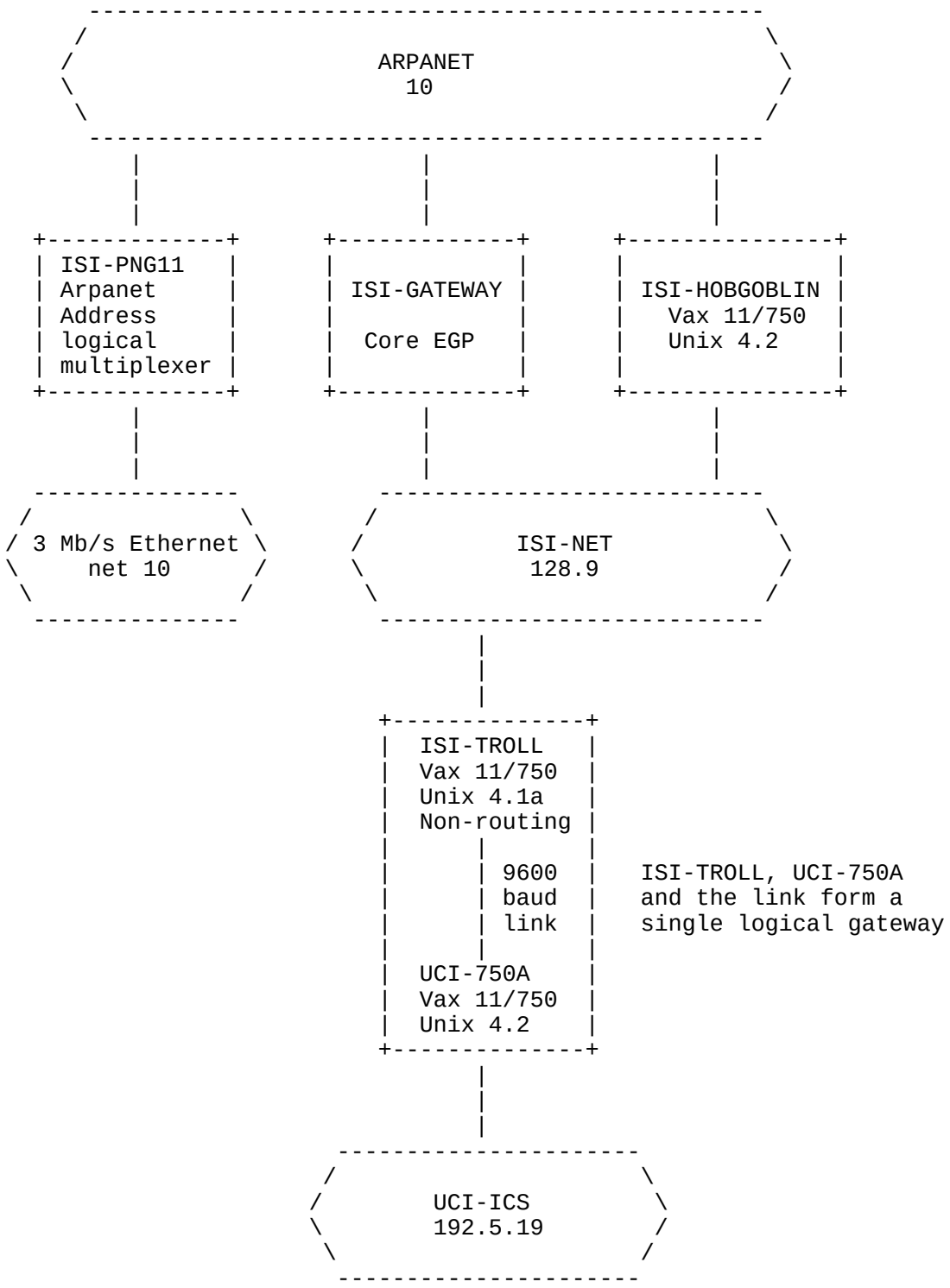

Figure 5-1: Simplified ISI Network Configuration

EGP can either be conducted with ISI-Gateway across ARPANET or ISI-NET.

5.2.1 EGP Across ARPANET

ISI-Hobgoblin will advise ISI-Gateway across ARPANET, and hence the core system, that it can reach ISI-NET and UCI-ICS.

Packets from AS's exterior to ISI and destined for UCI-ICS will be routed via ISI-Gateway, ISI-Hobgoblin and ISI-Troll. The extra hop via ISI-Gateway (or other core EGP gateway) is because the core gateways do not currently pass on indirect-neighbor exterior gateway addresses in their IGP messages (Gateway-to-Gateway Protocol). Packets originating from UCI-ICS destined for exterior AS's will be routed via ISI-Troll and ISI-Gateway. Thus the incoming and out going packet routes are different.

Packets originating from ISI-Hobgoblin as a host and destined for exterior AS's will be routed via the appropriate gateway on ARPANET.

UCI-ICS can only communicate with exterior AS's if ISI-Troll, ISI-Hobgoblin and ISI-Gateway are all up. The dependence on ISI-Gateway could be eliminated if ISI-Troll routed packets via ISI-Hobgoblin rather than ISI-Gateway. However, as ISI-Hobgoblin is primarily a host and not a gateway it is preferable that ISI-Gateway route packets when possible.

ISI-Hobgoblin can provide a back-up gateway function to ISI-Gateway as it can automatically switch to an alternative core EGP peer if ISI-Gateway goes down. Even though ISI-Hobgoblin normally advises the core system that it can reach ISI-NET the core uses its own internal route via ISI-Gateway in preference. For hosts on ISI-NET to correctly route outgoing packets they need their static gateway entries changed from ISI-Gateway to ISI-Hobgoblin. At present this would have to be done manually. This would only be appropriate if ISI-Gateway was going to be down for an extended period.

5.2.2 EGP Across ISI-NET

ISI-Hobgoblin will advise ISI-Gateway across ISI-NET that its indirect neighbor, ISI-Troll, can reach UCI-ICS net.

All exterior packet routing for UCI-ICS will be via ISI-Gateway in both directions with no hops via ISI-Hobgoblin. Packets originating from ISI-Hobgoblin as a host and destined for exterior AS's will be routed via ISI-Gateway, rather than the ARPANET interface, in both directions, thus taking an additional hop.

UCI-ICS can only communicate with exterior AS's if ISI-Troll and ISI-Gateway are up and ISI-Hobgoblin has advised ISI-Gateway of the UCI-ICS route. If ISI-Hobgoblin goes down, communication will still be possible because ISI-gateway (and other core gateways) do not time out routes to indirect

neighbors. If ISI-Gateway then goes down, it will need to be readvised by ISI-Hobgoblin of the UCI-ICS route, when it comes up.

Conducting EGP over ISI-NET rather than ARPANET should provide more reliable service for UCI-ICS for the following reasons: ISI-Gateway is specifically designed as a gateway, it is expected to be up more than ISI-Hobgoblin, it is desirable to eliminate extra routing hops where possible and, the exterior routing information will persist after ISI-hobgoblin goes down. If ISI-Hobgoblin is to be used in its back-up mode, EGP could be restarted across ARPANET after the new gateway routes are manually installed in the hosts. Therefore, EGP across ISI-NET was selected as the preferred mode of operation.

5.2.3 Potential Routing Loop

Because both ISI-Gateway and ISI-Hobgoblin provide routes between ARPANET and ISI-NET there is a potential routing loop. This topology in fact violates the original tree structure restriction. Provided ISI-Hobgoblin does not conduct EGP simultaneously with ISI-Gateway over ISI-NET and ARPANET, the gateways will only ever know about the alternative route from the shared EGP network and not from the other network. Thus a loop cannot occur. For instance, if EGP is conducted over ISI-NET, both ISI-Gateway and ISI-Hobgoblin will know about the alternative routes via each other to ARPANET from ISI-NET, but they will not know about the gateway addresses on ARPANET to be able to access ISI-NET from ARPANET. Thus they have insufficient routing data to be able to route packets in a loop between themselves.

5.3 Possible Future Configuration

5.3.1 Gateway to UCI-ICS

An improvement in the reliability and performance of the service offered to UCI-ICS can be achieved by moving the UCI-ICS interface from ISI-Troll to ISI-Hobgoblin. Reliability will improve because the connection will only require ISI-Hobgoblin and its ARPANET interface to be up and performance will improve because the extra gateway hop will be eliminated.

This will also allow EGP to be conducted across ARPANET giving access to the alternative core gateways running EGP. This will increase the chances of being able to reliably acquire an EGP neighbor at all times. It will also eliminate the extra hop via ISI-Gateway for packets originating from ISI-Hobgoblin, as a host, and destined for exterior networks.

This configuration change will be made at sometime in the future. It was not done initially because ISI-Hobgoblin was experimental and down more often than ISI-Troll.

5.3.2 Dynamic Switch to Backup Gateway

It was noted in Section 5.2.1 that ISI-Hobgoblin can provide a backup gateway function to ISI-Gateway between ARPANET and ISI-NET. Such backup gateways could become a common approach to providing increased reliability.

At present the change over to the backup gateway requires the new gateway route to be manually entered for hosts on ISI-NET. This section describes a possible method for achieving this changeover dynamically when the primary gateway goes down.

The aim is to be able to detect when the primary gateway is down and have all hosts on the local network change to the backup gateway with a minimum amount of additional network traffic. The hosts should revert back to the primary gateway when it comes up again.

The proposed method is for only the backup gateway to monitor the primary gateway status and for it to notify all hosts of the new gateway address when there is a change.

5.3.2.1 Usual Operation

The backup gateway runs a process which sends reachability-probe messages, such as ICMP echoes, to the primary gateway every 30 seconds and uses the responses to determine reachability as for EGP. If the primary gateway goes down a "gateway-address message" indicating the backup gateway address is broadcast (or preferably multicast) to all hosts. When the primary gateway comes up another gateway message indicating the primary gateway address is broadcast. These broadcasts should be done four times at 30 second intervals to avoid the need for acknowledgements and knowledge of host addresses.

Each host would run a process that listens for gateway-address messages. If a different gateway is advised it changes the default gateway entry to the new address.

5.3.2.2 Host Initialization

When a host comes up the primary gateway could be down so it needs to be able to determine that it should use the backup gateway. The host could read the address of the primary and backup gateways from a static initialization file. It would then set its default gateway as the primary gateway and send a "gateway-request message" to the backup gateway requesting the current gateway address. The backup gateway would respond with a gateway-address message. If no response is received the gateway-request should be retransmitted three times at 30 second intervals. If no response is received the backup gateway can be assumed down and the primary gateway retained as the default.

Whenever the backup gateway comes up it broadcasts a gateway-address message.

Alternatively, a broadcast (or multicast) gateway-request message could be

defined to which only gateways would respond. The backup gateway-address message needs to indicate that it is the backup gateway so that future requests need not be broadcast. Again, three retransmissions should be used. But the primary gateway also needs to broadcast its address whenever it comes up.

5.3.2.3 When Both the Primary and Backup are Down

If the primary gateway is down and the backup knows it is going down, it should broadcast gateway-address messages indicating the primary gateway in case the primary gateway comes up first.

But the backup could go down without warning and the primary come up before it. If the primary gateway broadcasts a gateway-address message when it comes up there is no problem. Otherwise, while hosts are using the backup gateway they should send a gateway-request message every 10 minutes. If no response is received it should be retransmitted 3 times at 30 second intervals and if still no response the backup assumed down and the primary gateway reverted to.

Thus the only time hosts need to send messages periodically is when the primary gateway does not send gateway-address messages on coming up and the backup gateway is being used. In some cases, such as at ISI, the primary gateway is managed by a different organization and experimental features cannot be conveniently added.

5.3.2.4 Unix 4.2 BSD

One difficulty with the above is that there is no standard method of specifying internet broadcast or multicast addresses. Multicast addressing is preferable as only those participating need process the message (interfaces with hardware multicast detection are available). In the case of Unix 4.2 BSD an internet address with zero local address is assumed for the internet broadcast address. However, the general Internet Addressing policy is to use an all ones value to indicate a broadcast function.

On Unix 4.2 BSD systems, both the gateway and host processes could be run at the user level so that kernel modifications are not required.

A User Datagram Protocol (UDP) socket could be reserved for host-backup-gateway communication.

Super user access to raw sockets for sending and receiving ICMP Echo messages requires a minor modification to the internet-family protocol switch table.

6. ACKNOWLEDGEMENT

I acknowledge with thanks the many people who have helped me with this project, but in particular, Dave Mills, who suggested the project, Jon Postel for discussion and encouragement, Liza Martin for providing the initial EGP code, Berkeley for providing the "routed" code, Mike Brescia for assistance with testing, Telecom Australia for funding me, and ISI for providing facilities.

7. REFERENCES

[Berkeley 83] "Unix Programmer's Manual", Vol. 1, 4.2 Berkeley Software Distribution, University of California, Berkeley.

[Kirton 84] Kirton, P.A., "EGP Gateway Under Berkeley Unix 4.2", University of Southern California, Information Sciences Institute, Research Report ISI/RR-84-145, to be published.

[Mills 83] Mills, D.L., "EGP Models and Self-Organizing Systems" Message to EGP-PEOPLE@BBN-UNIX, Nov. 1983.

[Mills 84a] Mills, D.L., "Exterior Gateway Protocol Formal Specification", Network Information Center RFC 904, April 1984.

[Mills 84b] Mills, D.L., "Revised EGP Model Clarified and Discussed", Message to EGP-PEOPLE@BBN-UNIX, May 1984.

[Postel 84] Postel, J., "Exterior Gateway Protocol Implementation Schedule" Network Information Center RFC 890, Feb. 1984.

[Rose 84] Rose, M.T., "Low-Tech Connection into the ARPA-Internet: The Raw-Packet Split Gateway", Department of Information and Computer Science, University of California, Irvine, Technical Report 216, Feb. 1984.

[Rosen 82] Rosen, E.C., "Exterior Gateway Protocol", Network Information Center RFC 827, Oct. 1982.

[Seamonson & Rosen 84] Seamonson, L.J. and Rosen, E.C., "Stub Exterior Gateway Protocol", Network Information Center RFC 888, Jan. 84.

[Xerox 81] "Internet Transport Protocols", Xerox System Integration Standard XSIS 028112, Dec. 1981.