              A MORE FAULT TOLERANT APPROACH TO ADDRESS RESOLUTION FOR
                       A MULTI-LAN SYSTEM OF ETHERNETS

STATUS OF THIS MEMO

   This memo discusses an extension to a Bridge Protocol to detect and
   disclose changes in neighbouring host address parameters in a Multi-
   LAN system of Ethernets. The problem is one which is appearing more
   and more regularly as the interconnected systems grow larger on
   Campuses and in Commercial Institutions. This RFC suggests a
   protocol enhancement for the Internet community, and requests
   discussion and suggestions for improvements. Distribution of this
   memo is unlimited.

ABSTRACT

   Executing a protocol P, a sending host S decides, through P's routing
   mechanism, that it wants to transmit to a target host T located
   somewhere on a connected piece of 10Mbit Ethernet cable which
   conforms to IEEE 802.3. To actually transmit the Ethernet packet, a
   48 bit Ethernet/hardware address must be generated. The addresses
   assigned to hosts within protocol P are not always compatible with
   the corresponding Ethernet address (being different address space
   byte orderings or values). A protocol is presented which allows
   dynamic distribution of the information required to build tables that
   translate a host's address in protocol P's address space into a 48
   bit Ethernet address. An extension is incorporated to allow such a
   protocol to be flexible enough to exist in a Transparent Bridge, or
   generic Host. The capability of the Bridge to detect host reboot
   conditions in a multi-LAN environment is also discussed, emphasising
   particularly the effect on channel bandwidth. To illustrate the
   operation of the protocol mechanisms, the Internet Protocol (IP) is
   used as a benchmark [6], [8]. Part 1 presents an introduction to
   Address Resolution, whilst Part 2 discusses a reboot detection
   process.

DEFINITIONS:

        CATENET: a group of IP networks linked together
        IP   : Internet Protocol

PART 1

INTRODUCTION

   In the Ethernet, while all packets are broadcast, the hardware
   interface selects only those with either the explicit hardware
   broadcast address or the individual hardware address of this
   interface. Packets which do not have one of these two addresses are
   rejected by the interface and do not get passed to the host software.
   This saves a great deal of otherwise wasted effort by the host
   software having to examine packets and reject them. If the interface
   hardware selected packets to pass to the host software by means of
   the protocol address, there would be no need for any translation from
   protocol to Ethernet address. Although it is very important to
   minimize the number of packets which each host must examine, so
   reducing especially needless inspections, use of the hardware
   broadcast address should be confined to those situations where it is
   uniquely beneficial. Perhaps if one were designing a new local
   network one could eliminate the need for an address translation, but
   in the real world of existing networks it fills a very important
   purpose. A rare use of the broadcast hardware address, which avoids
   putting any processing load on the other hosts of the Ethernet, is
   where hosts obtain the information they need to use the specific and
   individual hardware addresses to exchange most of their packets.

REASONING BEHIND ADDRESS RESOLUTION

   The process of converting from the logical host address to the
   physical Ethernet address has been termed ADDRESS RESOLUTION, and has
   prompted research into a method which can be easily interfaced,
   whilst at the same time remaining portable.

   The Ethernet requires 48 bit addresses on the physical cable [11] due
   to the fact that the manufacturers of the LAN interface controllers
   assign a unique 48 bit address during production. Of course, Network
   Managers do not want to be bothered using this address to identify
   the destination at the higher-level. Rather, they would prefer to
   assign their logical names to the hosts within their supervision, and
   allow some lower level protocol to perform a resolving operation.
   Most of these logical protocol addresses are not 48 bits long, nor do
   they necessarily have any relationship to the 48 bit address space.

   For example, IP addresses have a 32 bit address space [6], thus
   giving rise to the need to distribute dynamically the correspondences
   between a <PROTOCOLTYPE,PROTOCOL-ADDRESS> pair, and a 48 bit Ethernet
   address.

EXAMPLE ARP OPERATION

   Here is a review of the operation of ARP as defined in RFC-826 [5].
   Let hosts X and Y exist on the same Ethernet cable. They have
   physical Ethernet addresses EA(X), and EA(Y), and DoD Internet
   addresses IPA(X), and IPA(Y). Let the Ethernet type of Internet be
   ET(IP). Host X begins an application, and sooner or later wishes to
   communicate an Internet packet to host Y. Host X has knowledge of
   the Internet address of Y, i.e., (IPA(Y)), and informs the lower
   level that it wishes to talk to IPA(Y). The lower-level subsequently
   consults the ARP Module (ARM) to convert <ET(IP),IPA(Y)> into a 48
   bit Ethernet address but because X has not talked to Y previously, it
   does not have this information in its Translation Cache (TC). It
   discards (or queues) the Internet packet, and creates a new Address
   Resolution packet with:

        PACKET FIELD   VALUE ASSIGNED

         HRDTYP   ETHERNET

         PROTYP   ET(IP)

         HRDLEN   length (EA(X))

         PROTLEN   length (IPA(X))

         ARPOPC   REQUEST

         SOURCE HWR   EA(X)

         SOURCE PROT   IPA(X)

         TARGET HWR   don't know

         TARGET PROT   IPA(Y)

   It then broadcasts this packet to all hosts on the connecting cable.
   Host Y picks up this packet and determines that it understands the
   hardware type (Ethernet), that it speaks the indicated protocol
   (Internet), and that the packet is for it, that is, TARGET PROTOCOL
   ADDRESS = IPA(Y). Replacing any previous entry, it enters the
   information that <ET(IP),IPA(X) translates to EA(X). It then learns
   that this is an ARREQ packet, so it swaps fields, placing EA(Y) in
   the new sender Ethernet address field SOURCE HARDWARE ADDRESS, EA(X)
   as TARGET HARDWARE ADDRESS, IPA(X) as TARGET PROTOCOL ADDRESS, IPA(Y)
   as SOURCE PROTOCOL ADDRESS, and sets the opcode to REPLY. The packet
   is then sent with direct routing address information to EA(X). Thus,
   Y now knows how to send to X, but X still doesn't know EA(Y).

When X receives the ARREP packet from Y, it gets the address
information into its translation cache ET(IP),IPA(Y)>-->EA(Y),
notices that it is a REPLY, and discards the packet (i.e., disposes
of the dynamic packet buffer). However, if the original Internet
Module packet had been queued, it could have been accessed and given
the full addressing information from the translation cache.
Alternatively, had it been discarded, the higher level would have
succeeded on a subsequent attempt, and the Internet packet would be
transmitted immediately.

OBTAINING GREATER NETWORKING RANGE

There are many benefits to be gained in dividing a large multiuser
network into smaller, more manageable networks. These include : Data
Security; Overall Network Reliability; Performance Enhancement; not
to mention the most obvious: Greater Networking Range. In some
network technologies, cable length may be stipulated not to exceed a
certain range due to electrical limitations. By installing a Bridge,
this restriction is effectively eliminated. An important
consideration is the effect the induced Bridge delays will have on
the protocol timeouts in operation on each LAN/Subnet. Careful
analysis of upper bounds on timeouts would have to be made in order
to gain full benefit from the increased range. In the case of
Ethernet the following system parameters exist [11], [12]:

   - the bus bandwidth is 10Mbit/s

   - the maximum node-to-node cable length is 1500 m

   - the maximum point-to-point link cable length is 1000 m

   - the maximum number of repeaters between two nodes is two

   - the worst case end-to-end bus propagation delay is 22.5 us

   - the jam time after collision is 32bit

   - the minimum interframe time is 9.6 us

   - the slot size is 512 bit = 51.2 us

Once a decision has being taken to subnet, the resulting subLANs may
be connected by including a Bridge to link them together and
providing a protocol which makes the collection of subnets appear as
a single network. The basic idea of the Bridge providing 'repeater'
facilities would not suffice in this application. Moreover, the
Bridge would have to have further 'intelligence' to enable it to
select those packets which are destined for remote networks based on

the protocol address of the target host. Thereby preventing it from
forwarding packets needlessly that will not be accepted. If this
procedure was not adhered to, the channel bandwidth on the remote
networks would be inundated with packets, causing local valid traffic
to backoff and the efficiency of the respective networks to rapidly
decrease.

One problem fundamental to the operation of the Bridge is how it
discovers on which LAN a particular host is interfaced. If there are
only two LANs in the system, each will have a dedicated cache at the
Bridge, and when a packet is received at the particular interface,
the source host's address parameters are entered in the respective
LAN cache. However, when we consider a Multi-LAN environment, the
procedure becomes more complicated.

```
 ___
|
|-----h3
|                                              E4
|-----hq                   |-----------------------|
|              _    B1      |           |           |
|-----hx      | |          |           |           |
|-------------| |          |           |           |
|-----h1      |_|          |           |           |
|              |    h19     |           |           |
|              |    |       | |         | |         _____
|              |    |       | |   B3    -----|_____|  B4
|              |    |       |_|         |           |
|-----he       |------------------| E2  |_|         |
|              |                   |     |           |
|-----h5       |                   |     |           |
|              |                   |     |           |
|              ---       ---       |     |           |
---            | |       |-------  |     |           |
E1             | | B2    |         |     |           |
               | |---------------| |     |           |
               ---               |     |             |------------------------
                                 |     |           |                        |
                                 ---   |-----------|                        |
                                 E3                                         |
                                                                           |
```
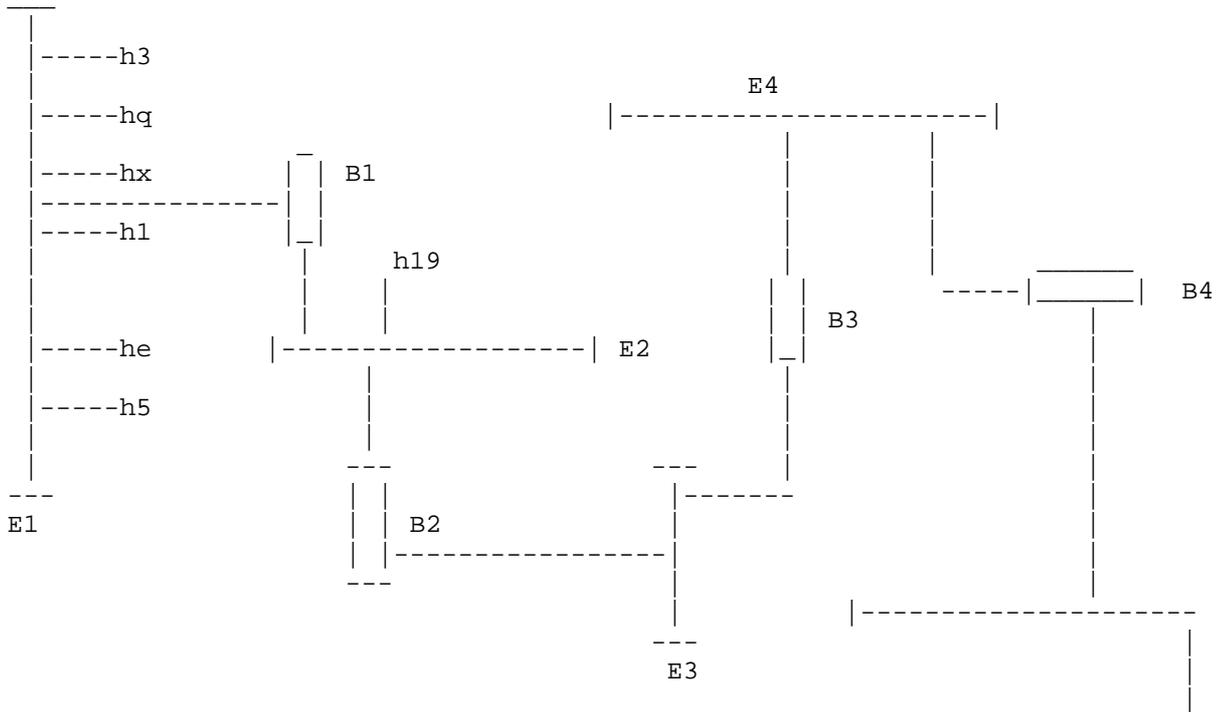
                    FIGURE 1. A MULTI-LAN TOPOLOGY

In the normal set-up, whenever B3 or B4 would receive a packet on E4,
they would both update the caches on their E4 interface. In
addition, a method must be provided to permit B4 to distinguish
between packets arriving on E4 from E1, E2, E3, and those which
actually originated on E4.

This is so that packets can be categorized as being of remote or
local source and processed accordingly. The most obvious solution is
for each Bridge to act as an AGENT and plug in its address as the
source of any packets it cascades to a remote network, instead of the
packet being cascaded with its original source address. At Bridge
boot, it may issue a broadcast request for all locally connected
hosts/devices to return their local network protocol addresses. On
subsequent receipt of this information, the Bridge could then update
the cache for each of its interfaces so that it would now have a base
from which to perform future operations.

The alternative to this automatic procedure is to permit manual
intervention in the Bridge software which could be activated by the
network manager in order to key in the addresses of the hosts
connected to each LAN interface.

Thus, having provided a means for the Bridge to obtain the original
state of the LAN addresses when it boots, how then does the Bridge
distinguish the arrival of a new host on the locally connected system
from transmissions which were sent from a remote source and cascaded
by an adjacent Bridge? Two approaches are currently under
consideration to solve this problem, namely Explicit Subnets, and
Transparent Subnets [4], [7], [9], [14].

In the Explicit Subnet approach, the location of the host in the
system is important. The address of the host in the protocol suite
will reflect which subnet the host is interfaced to. Consequently
the protocol address space is divided into a three level hierarchy of
<network,subnet,host>. Within the Internet there are five addressing
divisions in operation [10], classes A, B, C, D, and E. Classes D
and E relate to an addressing technique that will be used for
management of multi-casting groups and will not be discussed here.
With such a structure, it is possible to provide an address mask at
each interface so that received packets may have their source address
fields examined and compared with the address mask of this LAN. In
so doing, the component which is being verified is actually the
subnet address. If the masking operation is successful the source
must exist on this LAN, otherwise it must be remote.

With the Transparent scheme, the first time a newly booted host
'speaks' it will be looking for addressing information (probably
using BOOTSTRAP [1], RARP [2] or ARP [5]). Accordingly, the Bridge
will detect these respective requests and be in a position to perform
operations on the address parameters. The current approach in
Transparent Subnetting is that before any such requests can be
cascaded by the Bridge to an adjacent LAN, that Bridge will place its
interface address parameters into the source address fields, thus
acting as the AGENT. Therefore, this Bridge will 'see' either

packets arriving from the remote Bridge address, or local packets.
By virtue of the RARP/ARP operation, which hosts perform when they
first come up, any hi-level packets received on to the network not
having the bridge address, and not having a mapping in the cache for
that LAN, can be considered as being remote.

Currently, there is a move toward the Transparent subnet proposal
originally described by Postel [7]. This has been due mainly to
practical problems of incompatible implementations from different
vendors, and the restrictions that the Explicit address space place
on the adaptability of the system to change (class C addresses are
not flexible enough for the Explicit scheme). It is also the opinion
of the Author of this paper that the Agent technique adopted by the
Bridges could have shortcomings in a dynamic environment which would
be detrimental to its operation; for example, where the bridges
themselves relocate or crash, or in the management of the "Agent For
Who" cache at the bridge. Insofar as Loop Resolution and
SelfStabilization after failure are Bridge problems that need to be
addressed, it is strongly felt their satisfactory solution will be
supported by elimination of the Agent technique [13].

BRIDGE OPERATIONS

Referring to figure 1, assume that at some stage during its
processing [E1H3] wishes to communicate with [E2H19]. [E1H3] obtains
knowledge of the Internet address of [E2H19] from its translation
cache, but will not require the knowledge that [E2H19] exists on a
completely different subnet. [E1H3] calls its Internet Module to
transmit the packet. As detailed, the usual procedure of passing
control to its ARM is performed in an attempt to obtain a
translation. If we assume that [E1H3], and [E2H19] have not talked
before, the ARM in [E1H3] will not be able to resolve the addresses
on the first attempt.

In such a case, an ARREQ packet is assembled and broadcast to all
hosts on the network [E1]. The packet traverses the cable and is
eventually picked up by the (B1) Bridge Address Resolution Module
(BARM), whereupon it determines whether or not it should intervene in
the request. If the target is determined as remote (i.e., having no
match in the local cache), the BARM examines its Global Translation
Cache (GTC) to determine if it has an entry for <protocol,[E2H19]>.
Should a mapping be obtained at the Bridge, there is no need for the
broadcast REQUEST packet to be cascaded on to the remote network
[E2]. It is therefore assumed that the entries in the GTC reflect
the most current addressing information. A match thus obtained, the
original ARREQ packet buffer is adapted as required and returned
directly to [E1H3] via the Bridges hardware interface IFE1.

On the other hand, should the Bridges' GTC have no information on
[E2H19], the BARM would have to perform the following steps:

    1. drop the current ARREQ from [E1H3],

    2. create its own ARREQ using the Bridge source addresses
        and copy the target_internet_addr from the original
        [E1H3] ARREQ packet,

    3. broadcast the ARREQ on network E2 via network interface
        IFE2, and go into a timeout awaiting a REPLY.

Should this timeout period expire, a number of retries will be
permitted under control of the BARM. Alternatively, if a REPLY is
received within the timeout interval, then the BARM will update its
GTC. The ARM of [E1H3] next will attempt to transmit another ARREQ,
but this time a mapping will be obtained at the BARM'S GTC, and the
appropriate REPLY will be returned.

Part 1 has described the state of the art of the behaviour of Address
Resolution. Part 2 now extends the study to the more serious problem
of rebooting hosts in a multi-LAN system of Ethernets, and the
effects such changes have on the integrity of state information held
in ARP caches and routing tables.

                              PART 2

THE CAPTURE OF REBOOTS

Because Address Resolution packets are broadcast, all hosts on the
connecting cable including the Transparent Bridge will pick them up
and determine what they are. Referring to figure 1, it may well be
the case that a host on E1 wishes to communicate with a fellow host
on the same physical ether. Hence, if Hx wishes to talk to Hw on the
same ether, but has not done so previously, it will broadcast an
Address Resolution packet in the normal fashion. The Bridge will
also 'see' the packet as it passes by, and will act as described
above, unless that is, there is some method of preventing it doing
so; there is no point in the Bridge invoking its ARM, and wasting
processing time if the problem is going to be resolved locally.

It may occur however, that H1 wants to communicate with H5. If
however, H5 has not talked with anyone before (i.e., it has been
"dormant"), H1 will issue an ARREQ. The Bridge will not know that H5
is local because it won't have been entered in the local address
cache from previous conversations. To avoid broadcasting an ARREQ to
all networks/subnets, one way around this problem is to set up the
contents of the local cache at Bridge startup time. Therefore, the

Bridge will already know not to intervene. Thus, if the Bridge (with
2 nets) finds that a particular IP destination address is not in the
local cache of interface 1, it would have to examine its GTC and scan
it for a mapping. Should no mapping be obtained at interface 2, one
of two possibilities exist:

   1. the target host doesn't exist locally

   2. the caches are corrupt (the eventuality of this should
      be negligible!)

If it is assumed that each of the translation caches contains have
the most recent addressing information regarding its own domain of
the network then, in this example, if the Bridge does not get a
mapping at the GTC it would appear that the host must exist remotely
from E1, and E2.

Such a conclusion would ignore cases in which a host unplugs from a
particular hardware interface and plugs into another hardware
interface, or where logical names are reassigned to different
interfaces due to host user change. Either of these events could
happen had the host being accessed on E2, which would mean that a
REBOOT has taken place.

Anticipating these possiblities local caches are essential. In
normal operation, the Bridge will process and forward IP packets
received from one network, and destined for another. If the Bridge
picks up an ARREQ, it will first look for a mapping in its GTC before
discarding the original ARREQ, and transmitting its own to the remote
network. In any case, the Bridge will always examine the local cache
entries at the receiving interface, so that it may determine if the
target address is local or remote. When the Bridge first scans the
local cache, it does so with the source IP address as the key. If no
mapping is retrieved, it then scans the GTC with the same key.
Should a mapping now be obtained, it remains for the Bridge to insert
the source IP into the local cache, where it has either been
previously deleted or corrupted.

However, if the source IP exists in the respective local cache, the
validity of the source Ethernet address should also be verified by
examining the respective entry in the GTC. A scan of the GTC is then
performed with <protocol,source_prot_addr> as the key. If a mapping
is retrieved, the respective <et_addr> should be checked against the
source Ethernet address in the packet header. If the addresses do
not match, then we have uncovered a Hardware Reboot condition (i.e.,
a change in Ethernet ID). On the other hand, should the scan of the
GTC with <protocol,source_prot_addr> fail to obtain a mapping, then
the Bridge would scan the GTC with the current Ethernet address in

the packet header. If this obtains a mapping, then a Protocol Reboot
condition (i.e., change in logical ID) has been detected.

In the next section, the implications of these forms of 'Reboot' are
discussed.

REBOOT SCENARIO

In normal operation, packets will uneventfully traverse each subnet
either as complete Internet packets, broadcast ARREQ's, or direct
ARREP's. The Bridge attached to each subnet will 'hear', and 'see'
all packets as they travel past its connected interfaces. Because of
the existence of the local caches at each interface, the Bridge can
decide whether or not to intervene. In general circumstances, each
host on the Catenet will have a translation cache containing
<protocol,source_prot_addr,source_et_addr> entries for all packets it
has observed. Most of these entries will have been due to processing
ARREQ packets, which were broadcast, and by receiving REPLY packets.
In accordance with the foregoing , the Bridge will have a cache
attached to each subnet interface containing entries for protocol
addresses.

Within the Bridge's Global Translation Cache (GTC) will be entries of
all <protocol,source_prot_addr,source_hrd_addr> triplets relating to
valid hosts which have been recognised. If we assume that we have
just connected up a Catenet such as that illustrated in figure 1,
then at power-up no stations will have knowledge about their
neighbours. If the Bridges are to remain transparent, the
translation caches at each host will be totally empty. The only
addressing details that will be in existence will be the protocol
addresses stored in the local caches of the Bridges.

The hosts subsequently begin to run applications and will want to
communicate with one another. The first ARREQ is broadcast on the
respective subnet and all hosts, including the Bridge's interface to
the subnet, will pick it up and store the details. If, for example,
Hx issues an ARREQ for Hq, the Bridge will not intervene since there
is no need (providing no reboot has occurred at Hq). However, if Hx
wishes to talk with Hz, B1 will determine that the target IP in the
respective ARREQ does not exist in the local cache of IFE1, so it
will examine the GTC, with the <protocol,target_prot_addr> of Hw as
the key.

It is assumed that there will be a timeout mechanism in operation at
the source of any packet. In addition, the Bridge may also place the
target address in a 'search list' of currently sought hosts, so as to
prevent ARREQs from different sources being cascaded for the same
target. Under these conditions, Hx may re-issue its original ARREQ,

   but will be ignored until the host Hw has replied to the ARREQ
   transmitted by the Bridge.

NORMAL RUNNING STATE

   Assuming that a few ARP's have been issued, IP packets will start
   traversing the Catenet with full addressing information. Again, the
   Bridges will 'see' all the packets. If we extend the situation one
   step further, and assume that several conversations have taken place
   across the Catenet, there will be entries in the translation caches
   of the hosts concerned, regarding the
   <protocol,target_prot_addr,target_hrd_addr> triplets of those hosts
   with which the conversations took place. The Bridges also, will have
   details in their GTC's for packets which they cascaded.

   If a host is relocated, any connections initiated by that host will
   still work, provided that its own translation cache is cleared when
   it does physically move. However, any connections subsequently
   initiated to it by other hosts on the Catenet will have no particular
   reason to know to discard their old translation for that host.
   Ideally, 48 bit Ethernet addresses will be unique and fixed for all
   time.

RECOGNITION OF THESE REBOOT CONDITIONS

   With reference to figure 1, assume that for some reason a fault
   occurs on the hardware interface of <E1He>. The result of this is
   that a new interface is installed with a newly acquired hardware
   address. When <E1He> is powered up, the previous contents of its
   translation cache are cleared and it has no recollection of local, or
   remote host addresses. Accordingly, <E1He> begins to issue ARREQ's
   to hosts it requires. Whenever <E1He> transmits its first ARREQ, it
   could be termed a 'HELLO PACKET', since everyone on the subnet can
   pick up the packet, and store the relevant information in their
   translation caches. Within hosts, a mapping will be found on the old
   <protocol,source_prot_addr> pair, and the current <et_addr> of the
   packet header will replace whatever is entered in the translation
   cache.

   At this point it would be easy for each host with an entry to
   recognise the Hardware Reboot situation and inform the subnet with a
   respective broadcast reboot packet. But allowing such a procedure
   would be extremly inefficient on the broadcast medium, and would
   drastically outweigh any improvements in performance which might be
   obtained in the long term. In any case, given the fact that the
   ARREQ is broadcast, all stations on the subnet will recognise the
   reboot. The important point to consider is the effect such a reboot
   will have on subsequent conversations which are initiated remotely.

Can redundant transmissions be thwarted before they tie up processing
time on hosts en-route to the rebooted target? How these
difficulties are resolved is critical to the level of performance
obtained in a Catenet configuration. Since it is not optimal for
hosts to inform the system of a reboot, it is left to the Bridge.
Whenever the Bridge receives a packet, be it IP, or ARP, it examines
the source address parameters in the packet header, in the hope of
detecting any incompatibilities between them and the entries in its
caches. There are three distinct possibilities, namely, a difference
in the 48 bit hardware address only, a difference in the protocol
address, and two completely new addresses. If an incompatibility is
discovered, a "REBOOT" packet is constructed and issued on all remote
interfaces containing the appropiate information, allowing Bridges to
update their GTC's and generic hosts their ARP caches.

The structure of the Reboot packet is as depicted in figure 2.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| P A C K E T    O P C O D E  |REB OPC|       S O U R C E       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|        H A R D W A R E         A D D R E S S                  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|       S O U R C E   P R O T O C O L    A D D R E S S          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     M U L T I C A S T   T A R G E T   H A R D W A R E         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    A D D R E S S      |   M U L T I C A S T      T A R G E T   |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   P R O T O C O L     |
+-+-+-+-+-+-+-+-+-+-+-+-+

         --------> NEXT FOLLOWS A VARIANT FIELD ON REBOOT  OPCODE

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| O L D        S O U R C E      H A R D W A R E                 |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| A D D R E S S        |
+-+-+-+-+-+-+-+-+-+-+-+-+

 OR

+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| O L D    S O U R C E    P R O T O C O L     A D D R E S S |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
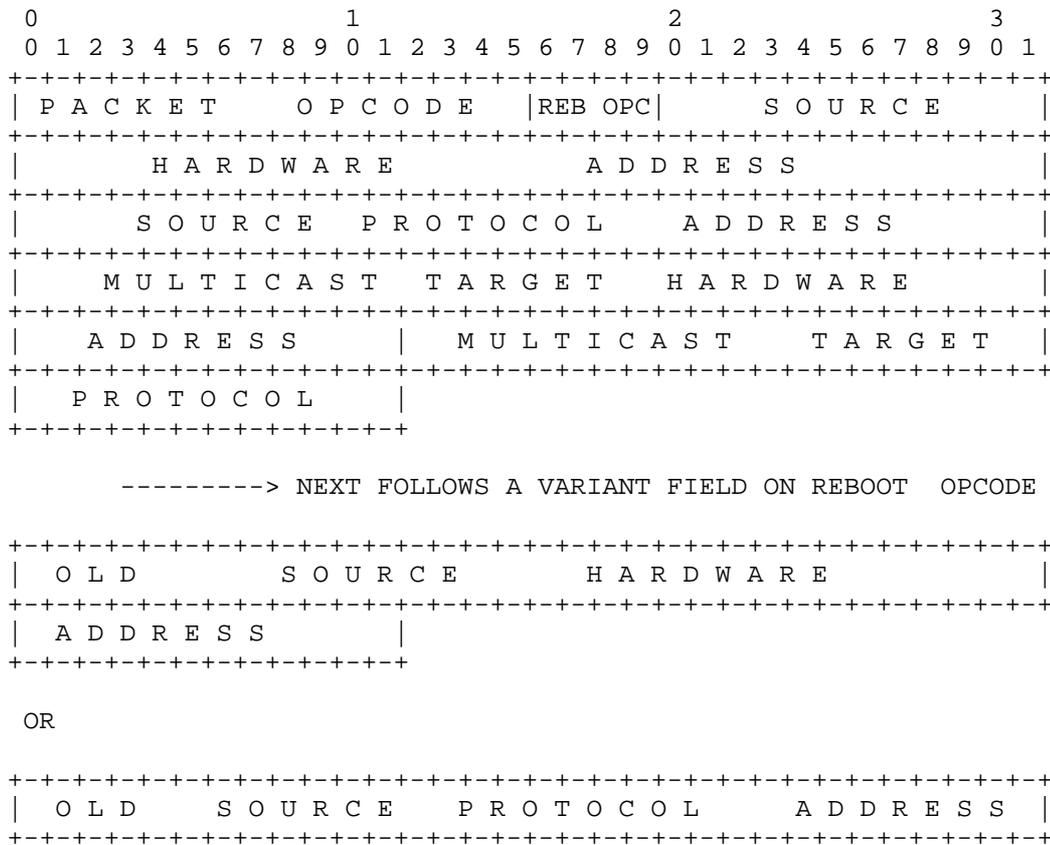
                   FIGURE 2. REBOOT PACKET

   The following definitions apply:

        PACKET FIELD   VALUE

        OPCODE   REBOOT

        REBOOT OPCODE   HARDWARE

        REBOOT OPCODE   PROTOCOL

   The format is then as follows:

        48 bit broadcast Ethernet address for the destination,

        48 bit Ethernet address of source Bridge,

        16 bit Protocol type = PACKET OPCODE - REBOOT.

   For completeness and error checking it may be an advantage to have a
   field which specifies the length of addresses in the Ethernet and
   protocol address spaces. Thus, the Reboot packet structure contains
   the following:

   FIELD   FIELD SIZE   DESCRIPTION

   HRDLEN   4 bit   byte length of Ethernet address

   PROTLEN   4 bit   byte length of Protocol address

   SOURCE
   PROTOCOL
   ADDRESS   32 bit   current protocol address of host

   TARGET
   PROTOCOL
   ADDRESS   32 bit   broadcast target protocol address

   REBOOT
   OPCODE   4 bit   will be either PROTOCOL or HARDWARE

   if   PROTOCOL   32 bit   old protocol address

   else HARDWARE   48 bit   old hardware address

As shown, depending on the REBOOT-OPCODE, the structure will continue
with either the 48 bit old hardware address or the 32 bit old
protocol address. The choice of a variant packet structure is for
reasons of curtailing the size of the packet to the fields that are
truely necessary in each situation. From this Reboot packet
structure, the process of generating such a packet can be considered.
When the Bridge algorithm detects a reboot, it should create a reboot
packet structure containing the relevant addressing information and
subsequently multicast it on the interface(s) which access(es) the
remote subnet(s). The decision as to which interface(s) is/are
local, and which is/are remote, can be resolved automatically
whenever a packet is received. With respect to this packet transfer
the receive interface at the Bridge becomes local, and all others are
tagged as remote.

Thus, hosts on the subnet remote from the reboot are informed of the
situation immediately as it is detected by the Bridge. In the
Catenet configuration illustrated in fig 1, this will have the effect
of updating the Translation Cache within each host, whenever it
receives the packet. If for example, <E4Hw> reboots under hardware,
B3 will detect this occurance. There is no reason for the subnets
E1, E2, E3 to be aware of this episode. In normal operation, B3 will
recognise the reboot from the first ARREQ issued from <E4Hw>. With
this reboot detection facility, B3 will be in a position to inform
the hosts on E1, E2, and E3. B3 can then create and issue the Reboot
packet via its interface with E3. When B3 picks it up, it will
update its own caches and subsequently cascade the packet onto E2,
where it will be passed on to E1 via B1.

ARGUMENTS FOR REBOOT PACKETS

It is envisaged that introducing Reboot packets, will serve to
enhance the bandwidth achievable within a Catenet system. Problems
of addressing 'dead' hosts will no longer exist in a correctly
functioning configuration. Translation Caches will have on hand the
most recent addressing information available, which should also serve
to enhance the performance of the routing strategy in operation.
Multiple, redundant processing of packets destined for 'dead' hosts
will be avoided. Weighing this against the processing involved with
a single multicast of Reboot packets, it is expected that the latter
will be is the most economically viable in relation to the long-term
traffic presented to the system.

CONCLUSION

It appears that reboots are becoming increasingly common on internet
networks. Many sites use Personal Computers (PC) as terminals and
the typical way to finish a session is to switch them off! With the

increasing popularity of multitasking Operating Systems on these
types of machines, problems are more likely to occur, particularly
when the PCs are diskless, or participating in a distributed file
system of some kind. Given the importance of correct addressing in
communications networks running Ethernet, it is anticipated the
reboot mechanism described will serve to improve the correctness and
validity of the protocol/network address mappings which may be stored
in the translation caches. To this degree, simulation is expected to
show that the volume of invalid traffic will decrease, to the benefit
of hosts, Bridges and servers alike. Likewise, ratification of the
routing policy is anticipated and since redundant/obsolete packets
will be thwarted, the efficient utilization of available channel
bandwidth across the catenet is also expected to improve. Thus,
effectively increasing Catenet throughput for 'valid' packets, and
therefore enhancing the level of service provided to the end users.

It is obvious that the proposed scheme implies the alteration of the
packet processing code in Bridges/Gateways. The point to remember is
the increased favour with which larger, more complex Multi-LAN
systems of Ethernets are being received. The recent adaption of
extra telephone cables to serve as the transmission media for the
Ethernet can only result in installation costs being reduced, therein
making the Ethernet more attractive within large corporate buildings,
etc. It is sensible to suggest that the probability of host address
re-assignment shall increase in proportion to the number of physical
systems attached, component failure rate (for whatever reason),
relocation of resources, and the size and turnover of the workforce
(i.e., people moving from one room to another). Simulation
experiments are currently being developed to analyse the resultant
traffic patterns under this scheme, and it is hoped to highlight
thresholds where adoption of the scheme becomes a necessity.

In addition, the Author is currently extending the boundaries of this
problem to encompass the reboot, or relocation of Bridges themselves.
Involved with this are the phenomena of loop resolution, load sharing
and duplicate packet suppression. It is envisaged that a Self-
Stabilizationg Bridge Protocol will result that will be more "light-
weight" than those adhering to the Spanning Tree Algorithm.

The Author would appreciate feedback/comments on this RFC. My
network address is: CBAD13%UCVAX.ULSTER.AC.UK@CUNYVM.CUNY.EDU.

ACKNOWLEDGEMENTS

REFERENCES

    [1]   Croft, Bill, and John Gilmore, "Bootstrap Protocol", RFC-951,
          Stanford University, September 1985.

    [2]   Finlayson, Mann, Mogul, and Theimer, "A Reverse Address
          Resolution Protocol", RFC-903, Computer Science Dept, Stanford
          University, June 1984.

    [3]   Lorimer, Alan, and Jim Reid, "ARP Information Communique",
          Computer Science Dept, Strathclyde University, 1987.

    [4]   Mogul, Jeffrey, "Internet Subnets", RFC-917, Computer Science
          Dept, Stanford University, October 1984.

    [5]   Plummer, David, "An Ethernet Address Resolution Protocol", RFC-
          826, MIT, November 1982.

    [6]   Postel, Jon, "DARPA Internet Program Protocol Specification",
          RFC-791, USC/Information Sciences Institute, September 1981.

    [7]   Postel, Jon, "Multi-LAN Address Resolution", RFC-925,
          USC/Information Sciences Institute, October 1984.

    [8]   Postel, Jon, Carl Sunshine, and Danny Cohen, "The ARPA Internet
          Protocol", Computer Networks, no. 5, pp. 261-271, 1981.

    [9]   Postel, Jon, and Jeff Mogul, "Internet Standard Subnetting
          Procedure", RFC-950, USC/Information Sciences Institute and
          Stanford University, August 1985.

    [10] Reynolds, Joyce, and Jon Postel, "Assigned Numbers", RFC-1010,
          USC/Information Sciences Institute, May 1987.

    [11] "The Ethernet: a local area network, data link layer and
          physical layer specification", Version 1.0 DEC, Intel and Xerox
          Corporations, USA 30 September 1980).

    [12] Hughes, H.D., and L. Li, "Simulation model of an Ethernet",
          Computer Performance, Vol 3, no. 4, December 1982.

    [13] Parr, Gerald P., "Address Resolution For An Intelligent
          Filtering Bridge Running On A Subnetted Ethernet System", ACM

SIGCOMM Computer Communication Review, (July/August 1987), vol.
17, no. 3.

[14] Smoot, Carl-Mitchell, and John S. Quarterman, "Using ARP to
Implement Transparent Subnet Gateways", RFC-1027, Texas Internet
Consulting, October 1987.