Network Working Group
Request for Comments: 1027

Smoot Carl-Mitchell
Texas Internet Consulting
John S. Quarterman
Texas Internet Consulting
October 1987

Using ARP to Implement Transparent Subnet Gateways

Status of this Memo

This RFC describes the use of the Ethernet Address Resolution Protocol (ARP) by subnet gateways to permit hosts on the connected subnets to communicate without being aware of the existence of subnets, using the technique of "Proxy ARP" [6]. It is based on RFC-950 [1], RFC-922 [2], and RFC-826 [3] and is a restricted subset of the mechanism of RFC-925 [4]. Distribution of this memo is unlimited.

Acknowledgment

The work described in this memo was performed while the authors were employed by the Computer Sciences Department of the University of Texas at Austin.

Introduction

The purpose of this memo is to describe in detail the implementation of transparent subnet ARP gateways using the technique of Proxy ARP. The intent is to document this widely used technique.

1. Motivation

The Ethernet at the University of Texas at Austin is a large installation connecting over ten buildings. It currently has more than one hundred hosts connected to it [5]. The size of the Ethernet and the amount of traffic it handles prohibit tying it together by use of repeaters. The use of subnets provided an attractive alternative for separating the network into smaller distinct units.

This is exactly the situation for which Internet subnets as described in RFC-950 are intended. Unfortunately, many vendors had not yet implemented subnets, and it was not practical to modify the more than half a dozen different operating systems running on hosts on the local networks.

Therefore a method for hiding the existence of subnets from hosts was highly desirable. Since all the local area networks supported ARP, an ARP-based method (commonly known as "Proxy ARP" or the "ARP hack") was chosen. In this memo, whenever the term "subnet" occurs the "RFC-950 subnet method" is assumed.

2. Design

2.1 Basic method

On a network that supports ARP, when host A (the source) broadcasts an ARP request for the network address corresponding to the IP address of host B (the target), host B will recognize the IP address as its own and will send a point-to-point ARP reply. Host A keeps the IP-to-network-address mapping found in the reply in a local cache and uses it for later communication with host B.

If hosts A and B are on different physical networks, host B will not receive the ARP broadcast request from host A and cannot respond to it. However, if the physical network of host A is connected by a gateway to the physical network of host B, the gateway will see the ARP request from host A. Assuming that subnet numbers are made to correspond to physical networks, the gateway can also tell that the request is for a host that is on a different physical network from the requesting host. The gateway can then respond for host B, saying that the network address for host B is that of the gateway itself. Host A will see this reply, cache it, and send future IP packets for host B to the gateway. The gateway will forward such packets to host B by the usual IP routing mechanisms. The gateway is acting as an agent for host B, which is why this technique is called "Proxy ARP"; we will refer to this as a transparent subnet gateway or ARP subnet gateway.

When host B replies to traffic from host A, the same algorithm happens in reverse: the gateway connected to the network of host B answers the request for the network address of host A, and host B then sends IP packets for host A to gateway. The physical networks of host A and B need not be connected to the same gateway. All that is necessary is that the networks be reachable from the gateway.

With this approach, all ARP subnet handling is done in the ARP subnet gateways. No changes to the normal ARP protocol or routing need to be made to the source and target hosts. From the host point of view, there are no subnets, and their physical networks are simply one big IP network. If a host has an implementation of subnets, its network masks must be set to cover only the IP network number, excluding the subnet bits, for the system to work properly.

2.2 Routing

As part of the implementation of subnets, it is expected that the elements of routing tables will include network numbers including both the IP network number and the subnet bits, as specified by the subnet mask, where appropriate. When an ARP request is seen, the ARP subnet gateway can determine whether it knows a route to the target host by looking in the ordinary routing table. If attempts to reach foreign IP networks are eliminated early (see Sanity Checks below), only a request for an address on the local IP network will reach this point. We will assume that the same network mask applies to every subnet of the same IP network. The network mask of the network interface on which the ARP request arrived can then be applied to the target IP address to produce the network part to be looked up in the routing table.

In 4.3BSD (and probably in other operating systems), a default route is possible. This default route specifies an address to forward a packet to when no other route is found. The default route must not be used when checking for a route to the target host of an ARP request. If the default route were used, the check would always succeed. But the host specified by the default route is unlikely to know about subnet routing (since it is usually an Internet gateway), and thus packets sent to it will probably be lost. This special case in the routing lookup method is the only implementation change needed to the routing mechanism.

If the network interfaces on which the request was received and through which the route to the target passes are the same, the gateway must not reply. In this case, either the target host is on the same physical network as the gateway (and thus the host should reply for itself), or this gateway is not on the most direct path to the desired network, i.e., there is another gateway on the same physical network that is on a more direct path and the other gateway should respond.

RFC-925 [4] describes a general mechanism for dynamic subnet routing using Proxy ARP and routing caches in the gateways. Our technique is restricted subset of RFC-925, in which we use static subnet routes which are determined administratively. As a result, our transparent subnet gateways require no new network routing table entries nor ARP cache entries; the only tables which are affected are the ARP caches in the host.

In our implementation, routing loops are prevented by proper administration of the subnet routing tables in the gateways.

2.3 Multiple gateways

The simplest subnet organization to administer is a tree structure, which cannot have loops. However, it may be desirable for reliability or traffic accommodation to have more than one gateway (or path) between two physical networks. ARP subnet gateways may be used in such a situation: a requesting host will use the first ARP response it receives, even if more than one gateway supplies one. This may even provide a rudimentary load balancing service, since if two gateways are otherwise similar, the one most lightly loaded is the more likely to reply first.

More complex mechanisms could be built in the form of gateway-to-gateway protocols, and will no doubt become necessary in networks with large numbers of subnets and gateways, in the same way that gateway-to-gateway protocols are generally necessary among IP gateways.

2.4 Sanity checks

Care must be taken by the network and gateway administrators to keep the network masks the same on all the subnet gateway machines. The most common error is to set the network mask on a host without a subnet implementation to include the subnet number. This causes the host to fail to attempt to send packets to hosts not on its local subnet. Adjusting its routing tables will not help, since it will not know how to route to subnets.

If the IP networks of the source and target hosts of an ARP request are different, an ARP subnet gateway implementation should not reply. This is to prevent the ARP subnet gateway from being used to reach foreign IP networks and thus possibly bypass security checks provided by IP gateways.

An ARP subnet gateway implementation must not reply if the physical networks of the source and target of an ARP request are the same. In this case, either the target host is presumably either on the same physical network as the source host and can answer for itself, or the target host lies in the same direction from the gateway as does the source host, and an ARP reply from the would cause a loop.

An ARP request for a broadcast address must elicit no reply, regardless of the source address or physical networks involved. If the gateway were to respond with an ARP reply in this situation, it would be inviting the original source to send actual traffic to a broadcast address. This could result in the "Chernobyl effect" wherein every host on the network replies to such traffic, causing network "meltdown".

2.5 Multiple logical subnets per physical network

The most straightforward way to assign subnet numbers is one to one with physical networks. There are, however, circumstances in which multiple logical subnets per physical network are quite useful. One of the more common is when it is planned that a group of workstations will be put on their own physical network but the gateway to the new physical network needs to be tested first. (A repeater might be used when the gateway was not usable). If a rule of one subnet per physical network is enforced, the addresses of the workstations must be changed every time the gateway is tested. If they may be assigned addresses using a new subnet number while they are still on the old physical network, no further address changes are needed.

To permit multiple subnets per physical network, an ARP subnet gateway must use the physical network interface, not the subnet number to determine when to reply to an ARP request. That is, it should send a proxy ARP reply only when the source network interface differs from the target network interface. In addition, appropriate routing table entries for these "phantom" subnets must be added to the subnet gateway routing tables.

2.6 Broadcast addresses

There are two kinds of IP broadcast addresses: main IP directed network broadcast and subnet broadcast. An IP network broadcast address consists of the network number plus a well-known value in the rest (local part) of the address. An IP subnet broadcast is similar, except both the IP network number and the subnet number bits are included. RFC-922 standardized the use of all ones in the local part, but there were two conventions in use before that: all ones and all zeros. For example, 4.2BSD used all zeros, and 4.3BSD uses all ones. Thus there are four kinds of IP directed broadcast addresses still currently in use on many networks.

With transparent subnetting a subnet gateway must not issue an IP broadcast using the subnet broadcast address, e.g., 128.83.138.255. Hosts on the physical network that receive the broadcast will not understand such an address as a broadcast address, since they will not have subnets enabled (or will not have subnet implementations). In fact, 4.2BSD hosts (with or without subnet implementations) will instead treat an address with all ones in the local part as a specific host address and try to forward the packet. Since there is no such target host, there will be no entry in the forwarding host’s ARP tables and it will generate an ARP request for the target host. This presents the scenario (actually observed) of a 4.3BSD gateway running the rwho program, which broadcasts a packet once a minute,

causing every 4.2BSD host on the local physical network to generate an ARP request at the same time. The same problem occurs with any subnet broadcast address, whether the local part is all zeros or all ones.

Thus a subnet gateway in a network with hosts that do not understand subnets must take care not to use subnet broadcast addresses: instead it must use the IP network directed broadcast address instead.

Finally, since many hosts running out-of-date software will still be using (and expecting) old-style all-zeros IP network broadcast addresses, the gateway must send its broadcast addresses out in that form, e.g., 128.83.0.0. It might be safe to also send a duplicate packet with all ones in the local part, e.g., 128.83.255.255. It is not clear whether the local network broadcast address of all ones, 255.255.255.255, will cause ill effects, but it is very likely that it will not be recognized by many hosts that are running older software.

3. Implementation in 4.3BSD

Subnet gateways using ARP have been implemented by a number of different people. The particular method described in this memo was first implemented in 4.2BSD on top of retrofitted beta-test 4.3BSD subnet code, and has since been reimplemented as an add-on to the distributed 4.3BSD sources. The latter implementation is described here.

Most of the new kernel code for the subnet ARP gatewaying function is in the generic Ethernet interface module, netinet/if_ether.c. It consists of eight lines in in_arpinput that perform a couple of quick checks (to ensure that the facility is enabled on the source interface and that the source and target addresses are on different subnets), call a new routine, if_subarp, for further checks, and then build the ARP response if all checks succeed. This code is only reached when an ARP request is received, and does nothing if the facility is not enabled on the source interface. Thus performance of the gateway should be very little degraded by this addition. (Performance of the requesting host should also be similar to the latter case, as the only difference there is between efficiency of the ARP cache and of the routing tables).

The routine if_subarp (about sixty lines) ensures that the source and target addresses are on the same IP network and that the target address is none of the four kinds of directed broadcast address. It then attempts to find a path to the target either by finding a network interface with the desired subnet or by looking in the

routing tables. Even if a network interface is found that leads to the target, for a reply to be sent the ARP gateway must be enabled on that interface and the target and source interfaces must be different.

The file netinet/route.c has a static routing entry structure definition added, and modifications of about eight lines are made to the main routing table lookup routine, rtalloc, to recognize a pointer to that structure (when passed by if_subarp) as a direction to not use the default route in this routing check. The processor priority level (critical section protection) around the inner routing lookup check is changed to a higher value, as the routine may now be called from network interface interrupts as well as from the internal software interrupts that drive processing of IP and other high level protocols. This raised processor priority could conceivably slow the whole kernel somewhat if there are many routing checks, but since the critical section is fast, the effect should be small.

A key kernel modification is about fifteen lines added to the routine ip_output in netinet/ip_output.c. It changes subnet broadcast addresses in packets originating at the gateway to IP network broadcast addresses so that hosts without subnet code (or with their network masks set to ignore subnets) will recognize them as broadcast addresses. This section of code is only used if the ARP gateway is turned on for the outgoing interface, and only affects subnet broadcast addresses.

A new routine, in_mainnetof, of about fifteen lines, is added to netinet/in.c to return the IP network number (without subnet number) from an IP address. It is called from if_subarp and ip_output.

Two kernel parameter files have one line added to each: net/if.h has a definition of a bit in the network interface structure to indicate whether subnet ARP gateways are enabled, and netinet/in.h refers to in_mainnetof.

In addition to these approximately 110 lines of kernel source additions, there is one user-level modification. The source to the command ifconfig, which is used to set addresses and network masks of network interfaces, has four lines added to allow it to turn the subnet ARP gateway facility on or off, for each interface. This is documented in eleven new lines in the manual entry for that command.

4. Availability

The 4.3BSD implementation is currently available by anonymous FTP (login anonymous, password guest) from sally.utexas.edu as pub/subarp, which is a 4.3BSD "diff -c" listing from the 4.3BSD sources that were distributed in September 1986.

This implementation was not included in the 4.3BSD distribution proper because U.C. Berkeley CSRG thought that that would reduce the incentive for vendors to implement subnets per RFC-950. The authors concur. Nonetheless, there are circumstances in which the use of transparent subnet ARP gateways is indispensable.

References

1. Mogul, J., and J. Postel, "Internet Standard Subnetting Procedure", RFC-950, Stanford University and USC/Information Sciences Institute, August 1985.

2. Mogul, J., "Broadcasting Internet Datagrams in the Presence of Subnets", RFC-922, Computer Science Department, Stanford University, October 1984.

3. Plummer, D., "An Ethernet Address Resolution Protocol or Converting Network Protocol Addresses to 48-bit Ethernet Addresses for Transmission on Ethernet Hardware", RFC-826, Symbolics, November 1982.

4. Postel, J., "Multi-LAN Address Resolution", RFC-925, USC/Information Sciences Institute, October 1984.

5. Carl-Mitchell, S., and J. S. Quarterman, "Nameservers in a Campus Domain", SIGCUE Outlook, Vol.19, No.1/2, pp.78-88, ACM SIG Computer Uses in Education, P.O. Box 64145, Baltimore, MD 21264, Spring/Summer 1986.

6. Braden, R., and J. Postel, "Requirements for Internet Gateways", RFC-1009, USC/Information Sciences Institute, June 1987.