Network Working Group
Request for Comments: 2075
Category: Experimental

C. Partridge
BBN

January 1997

IP Echo Host Service

Status of this Memo

This memo defines an Experimental Protocol for the Internet community. This memo does not specify an Internet standard of any kind. Discussion and suggestions for improvement are requested. Distribution of this memo is unlimited.

Abstract

This memo describes how to implement an IP echo host. IP echo hosts send back IP datagrams after exchanging the source and destination IP addresses. The effect is that datagrams sent to the echo host are sent back to the source, as if they originated at the echo host.

Introduction

An IP echo host returns IP datagrams to their original source host, with the IP source and destination addresses reversed, so that the returning datagram appears to be coming from the echo host to the original source. IP echo hosts are tremendously useful for debugging applications and protocols. They allow researchers to create looped back conversations across the Internet, exposing their traffic to all the vagaries of Internet behavior (congestion, cross traffic, variable round-trip times and the like) without having to distribute prototype software to a large number of test machines.

IP echo hosts were heavily used on the Internet in the late 1970s and early 1980s to debug various Internet transport and application protocols. But, for reasons unclear, at the current date there are no echo hosts on the Internet and few people are even aware of the concept. The goal of this memo is to document the concept in the hopes it will be revived.

Implementation Details

While the basic idea of a echo host is simple, there are a few implementation details that require attention. This section describes those implementation details. The presentation works from the simplest to most difficult issues.

The most straightforward situation is when an echo host receives an IP datagram with no options and whose protocol field has a value other than 1 (ICMP). In this case, the echo host modifies the header by exchanging the source and destination addresses, decrements the TTL by one and updates the IP header checksum. The host then transmits the updated IP datagram back to the original source of the datagram.

NOTE: If the TTL is zero or less after decrementing, the datagram MUST not be echoed. In general, an echo host is required to do all the various sanity checks that a router or host would do to an IP datagram before accepting the datagram for echoing (see STD 3, RFC 1122, and RFC 1812).

The TTL MUST be decremented for security reasons noted below. Observe, however, that the effect is that hosts using an echo path through an echo host SHOULD set their TTL to twice the normal value to be sure of achieving connectivity over the echo path.

If an arriving IP datagram has options, the echo host's responsibilities are more complex. In general, the IP source and destination are always exchanged and TTL and checksum updated, but in certain situations, other special actions may have to take place.

If the datagram contains an incomplete source route option (i.e. the echo host is not the final destination), the datagram MUST be discarded. If the datagram contains a complete source route option, the source route option MUST be reversed, and the datagram (with source and destination IP addresses exchanged and updated TTL) MUST be sent back along the reverse source route.

More generally, the goal with any option is to update the option such that when the echoed packet is received at the original source, the option fields will contain data which makes sense for a datagram originating at the echo host.

There is one option for which it is unclear what the correct action. The timestamp option is sometimes used for round-trip time estimation. If the option is reset at the echo host, then a history of roughly half of the trip delay will be lost. But if the option is not reset, then the timestamp option will appear inconsistent with the source and destination addresses of the datagram. To try to balance these two issues, the following rules are suggested:

1. If the first entry in the timestamp option contains the IP address of the source host, the entry SHOULD be rewritten to contain the IP address of the echo host, and the timestamp option pointer SHOULD be truncated so that this timestamp is the only one

in the list. (This rewrite makes the option appear consistent with the new source and destination IP addresses, and retains the source timestamp, while losing information about the path to the echo host).

2. If the first entry in the timestamp option does not contain the IP address of the source host, the entry SHOULD be echoed back unchanged. The echo host SHOULD NOT appear in the timestamp option. (This approach retains the entire history of the path, though observe that on a symmetric route, it means every router may appear twice in the path).

Finally, if the IP datagram contains an ICMP packet (i.e. the IP protocol field value is 1), the datagram SHOULD be discarded. The reason for this rule is that the most likely reason for receiving an ICMP datagram is that an echoed datagram has encountered a problem at some router in the path and the router has sent back an ICMP datagram. Echoing the ICMP datagram back to the router may confuse the router and thus SHOULD be avoided. (This rule simply follows the Internet maxim of being conservative in what we send).

However, in some cases the ICMP datagram will have useful information for the source host which it would be desirable to echo. A sophisticated echo host MAY choose to echo ICMP datagrams according to the following rules:

1. Any ICMP datagram in which the destination address in the encapsulated IP header (the header within the ICMP datagram) matches the source address of the ICMP datagram MAY be safely echoed.

2. ICMP Source Quench and ICMP Destination Unreachable with a code of 4 (fragmentation needed and DF set) MAY be sent to the *destination* of the encapsulated IP datagram if the source IP address of the encapsulated IP datagram is that of the echo host. When the ICMP message is sent on, it SHOULD be rewritten as an ICMP message from the echo host to the source.

3. All other ICMP messages MUST be discarded.

These rules were chosen to try to ensure that end-to-end ICMP messages are passed through, as are messages from routers which are fairly safe and useful (or necessary) to the end system, but that potentially dangerous messages such as Redirects are suppressed. (The ICMP Destination Unreachable with code 4 is required for MTU discovery under RFC-1191).

Security Considerations

Echo hosts pose a number of security concerns related to address spoofing.

First, echo hosts provide obvious ways to extend attacks that make use of address spoofing. A malevolent host can write an third party's IP address as the source address of a datagram sent to an echo host and thus cause the echo host to send a datagram to the third party. In general, this trick does not create a new security hole (the malevolent host could just as well have sent the datagram with a forged source address straight to the third party host). But there are some new twists to the problem.

One exception is if the echo host is a host inside a firewall that accepts datagrams from hosts outside the firewall. In that case, a malevolent host outside the firewall may be able to use the echo host to make its packets appear to originate from inside the firewall (from the echo host). In general, a good firewall will catch these cases (the source address of the datagrams sent to the echo host will be for a host inside the firewall and testing for interior source addresses on datagrams arriving at an exterior interface is a standard firewall filter) but since the primary purpose of echo hosts is for wide scale Internet testing, there seems no reason to invite danger. So we recommend that echo hosts SHOULD NOT be placed inside firewalls.

Second, address spoofing can be used to cause flooding of the network. In this case, a malevolent host sends a datagram to an echo host with the source address of another echo host. This trick will cause datagrams to circulate between the two echo hosts. The requirement that the echo host decrement the TTL by one ensures that each datagram will eventually die, but a sufficiently malevolent host sending a large number of datagrams with high TTLs to an echo host can cause considerable disruption. There are a number of possible ways to repair this problem (such as requiring sources to authenticate themselves before sending datagrams to be echoed). A simple protection is simply to limit the number of packets echoed back to any one source per second. For instance, one might limit a source to a packet rate equal to 10% of the interface bandwidth (for a 10 Mb/s Ethernet this would be about 75 maximum sized packets per second).

One variation of this attack is to generate e-mail addressed to the echo host (e.g., user@echo.xxx.com). This e-mail will loop over the network a number of times until the SMTP server determines the message has too many Received-From: lines.

A third variation of the flooding trick is to place a multicast or broadcast address as the source of the IP datagram sent to an echo server. Since this results in an illegal arriving IP datagram, the echo server MUST discard the datagram. (This warning serves as a reminder that echo servers MUST do the standard checks for an illegal datagram before echoing).

Implementation Note

Echo hosts are often implemented as virtual interfaces on an existing host or router. One can think of the echo host's IP address as a second IP address for the host, with the semantics that all datagrams sent to that address get echoed. Observe that when an echo host is supported as a module within a larger host implementation, an easy implementation mistake to make is to accidentally put the non-echo address of a host into an echoed packet. For a variety of reasons (including security and correct operation of echo paths) implementors MUST ensure this NEVER happens.

Acknowledgements

This memo was stimulated by a conversation with Jon Crowcroft in which we both lamented the demise of some beloved IP echo hosts (e.g., goonhilly-echo.arpa). It has been considerably improved by comments from various members of the End2End-Interest mailing list, including Bob Braden, Mark Handley, Christian Huitema, Dave Mills, Tim Salo, Vern Schryver, Lansing Sloan, and Rich Stevens.

The author is emphatically not the inventor of echo hosts. Enquiries to the usual suspects suggest that echo hosts were created by persons unknown (probably at BBN) very early in the development of IP. I'd like to thank those persons who created echo hosts and apologize for any errors in describing their invention.

Author's Address

Craig Partridge
BBN Corporation
10 Moulton St
Cambridge MA 02138

EMail: craig@bbn.com