

v6ops
Internet-Draft
Intended status: Informational
Expires: February 25, 2015

M. Byerly
Fastly
M. Hite
Evernote
J. Jaeggli
Fastly
August 24, 2014

Close encounters of the ICMP type 2 kind (near misses with ICMPv6 PTB)
draft-v6ops-pmtud-ecmp-problem-00

Abstract

This document calls attention to the problem of delivering ICMPv6 type 2 "Packet Too Big" (PTB) messages to intended destinations in ECMP load balanced, anycast network architectures. It discusses operational mitigations that can address this class of failure.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of [BCP 78](#) and [BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at <http://datatracker.ietf.org/drafts/current/>.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on February 25, 2015.

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of

Internet-Draft

misses with ICMPv6 PTB

August 2014

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	2
2.	Problem	2
3.	Mitigation	4
3.1.	Alternatives	5
3.2.	Implementation	5
4.	Improvements	6
5.	Acknowledgements	7
6.	IANA Considerations	7
7.	Security Considerations	7
8.	Informative References	7
	Authors' Addresses	7

[1.](#) Introduction

Operators of popular Internet services face unique challenges associated with scaling their infrastructure. One approach is to utilize equal-cost multi-path (ECMP) routing to perform stateless distribution of incoming TCP or UDP sessions to multiple servers or middle boxes such as load balancers. Distribution of traffic in this manner presents a problem when dealing with ICMP signaling. Specifically, an ICMP error is not guaranteed to hash via ECMP to the same destination as its corresponding TCP or UDP session. A case where this is particularly problematic operationally is path MTU discovery (PMTUD).

[2.](#) Problem

A common application for stateless load balancing of TCP or UDP flows is to perform an initial subdivision of flows in front of a stateful load balancer tier or multiple servers so that the workload becomes divided into manageable fractions of the total number of flows. The flow division is performed using ECMP forwarding and a stateless but sticky algorithm for hashing across the available paths. This nexthop selection for the purposes of flow distribution is a constrained form of anycast d where all anycast destinations are equidistant topologically from the upstream router responsible for making the last next-hop forwarding decision before the flow arrives on the destination device. In this approach, the hash is performed

across some set of available protocol headers. Typically, these headers may include (IPv6)Flow-Label, IP-source, IP-destination, protocol, source-port, destination-port and potentially others such as ingress interface.

A problem common to this approach of distribution through hashing is impact on path MTU discovery. An ICMPv6 type 2 PTB message generated on an intermediate device for a packet sent from an ECMP load balanced server to a client, will have the load-balanced anycast address as the destination and will be statelessly load balanced to one of the anycast servers. While the ICMPv6 PTB message contains as much of the packet that could not be forwarded as possible, the payload headers do not factor into the forwarding decision and are ignored. Because the PTB message is not identifiable as part of the original flow by the packet header the results of the ICMPv6 ECMP hash are unlikely to be hashed to the same nexthop as packets matching TCP or UDP ECMP hash.

An example packet flow and topology follow.

```
ptb -> router ecmp -> nexthop L4/L7 load balancer -> destination

router --> load balancer 1 --->
      \\--> load balancer 2 ---> load-balanced service
      \--> load balancer N --->
```

Figure 1

The router ECMP decision is used because it is part of the forwarding architecture, can be performed at line rate, and does not depend on shared state or coordination across a distributed forwarding architecture which may include multiple linecards or routers. The ECMP routing decision is deterministic with respect to packets having the same computed hash.

The typical case where ICMPv6 PTB messages are received at the load balancer is a case where the path MTU from the client to the load balancer is limited by a tunnel in which the client itself is not aware of. In the common case of a TCP flow where TLS is employed,

the first packet that is likely to exceed a tunnel MTU lower than that specified by the MSS on the client and the load balancer/server is the TLS ServerHello and certificate.

Direct experience says that the frequency of PTB messages is small compared to total flows. One possible conclusion being that tunneled IPv6 deployments that cannot carry 1500 mtu packets are relatively rare. Techniques employed by clients such as happy-eyeballs may actually contribute some amelioration to the IPv6 client experience by preferring IPv4 in cases that might be identified as slow or failed. Still, the expectation of operators is that PMTUD should

work and that unnecessary breakage of client traffic should be avoided.

A final observation regarding server tuning is that it is typically not possible even if it is potentially desirable to be able to independently set the TCP MSS for different address families on end-systems.

The problem as described does also impact IPv4; however, the ability to fragment on wire at tunnel ingress points and the relative rarity of sub-1500 byte MTUs that are not coupled to changes in client behavior (for example, endpoint VPN clients set the tunnel interface MTU accordingly for performance reasons) makes the problem sufficiently rare that some deployments simply choose to ignore it.

[3.](#) Mitigation

Mitigation of the potential for PTB messages to be mis-delivered involves ensuring that an ICMPv6 error message is distributed to the same anycast server responsible for the flow for which the error is generated. Ideally Mitigation could be done by the mechanism hosts use to identify the flow, by looking into the payload of the ICMPv6 message (to determine which TCP flow it was associated with) before making a forwarding decision. Because the encapsulated IP header occurs at a fixed offset in the icmp message it is not outside the realm of possibility that routers with sufficient header processing capability could parse that far into the payload. Employing a mediation device that handles the parsing and distribution of PTB messages after policy routing or on each load-balancer/server is a

possibility.

Another mitigation approach is predicated upon distributing the PTB message to all anycast servers under the assumption that the one for which the message was intended will be able to match it to the flow and update the route cache with the new MTU, devices not able to match the flow will discard these packets. Such distribution has potentially significant implications for resource consumption and the potential for self-inflicted denial-of-service if not carefully employed. Fortunately, in real-world-deployment we have observed that, the number of flows for which this problem occurs is relatively small (example, 10 or fewer pps on 1Gb/s or more worth of https traffic) and sensible ingress rate limiters which will discard excessive message volume can be applied to protect even very large anycast server tiers with the potential for fallout only under circumstances of deliberate duress.

[3.1.](#) Alternatives

As an alternative, it may be appropriate to lower the TCP MSS to 1220 in order to accommodate 1280 byte MTU. We consider this undesirable as hosts may not be able to independently set TCP MSS by address-family thereby impacting IPv4, or alternatively that it relies on a middle-box to clamp the MSS independently from the end-systems.

[3.2.](#) Implementation

1. Filter-based-forwarding matches next-header ICMPv6 type-2 and matches a next-hop on a particular subnet directly attached to both border routers. The filter is policed to reasonable limits (we chose 1000pps).
2. Filter is applied on input side of all external interfaces
3. A proxy located at the next-hop forwards ICMPv6 type-2 packets received at the next-hop to an Ethernet broadcast address (example ff:ff:ff:ff:ff:ff) on all specified subnets. This was necessitated by router inability (in IPv6) to forward the same packet to multiple unicast next-hops.

4. Anycast servers receive the PTB error and process packet as needed.

A simple Python scapy script that can perform the ICMPv6 proxy reflection is included.

```
#!/usr/bin/python

from scapy.all import *

IFACE_OUT = ["p2p1", "p2p2"]

def icmp6_callback(pkt):
    if pkt.haslayer(IPv6) and (ICMPv6PacketTooBig in pkt) \
    and pkt[Ether].dst != 'ff:ff:ff:ff:ff:ff':
        del(pkt[Ether].src)
        pkt[Ether].dst = 'ff:ff:ff:ff:ff:ff'
        pkt.show()
        for iface in IFACE_OUT:
            sendp(pkt, iface=iface)
```

```
def main():
    sniff(prn=icmp6_callback, filter="icmp6 \
    and (ip6[40+0] == 2)", store=0)

if __name__ == '__main__':
    main()
```

This example script listens on all interfaces for IPv6 PTB errors being forwarded using filter-based-forwarding. It removes the existing Ethernet source and rewrites a new Ethernet destination of the Ethernet broadcast address. It then sends the resulting frame out the p2p1 and p2p2 interfaces where our anycast servers reside.

[4.](#) Improvements

There are several ways that improvements could be made to the situation with respect to ECMP load balancing of ICMPv6 PTB.

1. Routers with sufficient capacity within the lookup process could parse all the way through the L3 or L4 header in the ICMPv6 payload beginning at bit offset 32 of the ICMP header. By reordering the elements of the hash to match the inward direction of the flow, the PTB error could be directed to the same next-hop as the incoming packets in the flow.
2. The FIB could be programmed with a multicast distribution tree that included all of the necessary next-hops.
3. Ubiquitous implementation of [RFC 4821](#) [[RFC4821](#)] Packetization Layer Path MTU Discovery would probably go a long way towards reducing dependence on ICMPv6 PTB.

[5.](#) Acknowledgements

The authors would like to thank Mark Andrews, Brian Carpenter, Nick Hilliard and Ray Hunter, for review.

[6.](#) IANA Considerations

This memo includes no request to IANA.

7. Security Considerations

The employed mitigation has the potential to greatly amplify the impact of a deliberately malicious sending of ICMPv6 PTB messages. Sensible ingress rate limiting can reduce the potential for impact; however, legitimate traffic may be lost once the rate limit is reached.

The proxy replication results in devices not associated with the flow that generated the PTB being recipients of an ICMPv6 message which contains a fragment of a packet. This could arguably result in information disclosure. Recipient machines should be in a common administrative domain.

8. Informative References

[RFC4821] Mathis, M. and J. Heffner, "Packetization Layer Path MTU Discovery", [RFC 4821](#), March 2007.

Authors' Addresses

Matt Byerly
Fastly
Kapolei, HI
US

Email: mbyerly@zynga.com

Matt Hite
Evernote
Redwood City, CA
US

Email: mwhite@hotmail.com

Fastly
Mountain View, CA
US

Email: joelja@gmail.com