Abstract

Path MTU discovery remains widely undeployed due to security issues, and IP fragmentation has exposed weaknesses in application protocols. Currently, DNS is known to be the largest user of IP fragmentation. It is possible to avoid IP fragmentation in DNS by limiting response size where possible, and signaling the need to upgrade from UDP to TCP transport where necessary. This document proposes to avoid IP fragmentation in DNS.

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 https://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 March 30, 2020.

Copyright Notice

Copyright (c) 2019 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 (https://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
1. Introduction

DNS has EDNS0 [RFC6891] mechanism. It enables that DNS server can send large size response using UDP. Now EDNS0 is widely deployed, and DNS (over UDP) is said to be the biggest user of IP fragmentation.

However, "Fragmentation Considered Poisonous" [Herzberg2013] proposed effective off-path DNS cache poisoning attack vectors using IP fragmentation. "IP fragmentation attack on DNS" [Hlavacek2013] and "Domain Validation++ For MitM-Resilient PKI" [Brandt2018] proposed that off-path attackers can intervene in path MTU discovery [RFC1191] to perform intentionally fragmented responses from authoritative servers. [RFC7739] stated security implications of predictable fragment identification values.

And more, Section 3.2 Message Side Guidelines of UDP Usage Guidelines [RFC8085] specifies that an application SHOULD NOT send UDP datagrams that result in IP packets that exceed the Maximum Transmission Unit (MTU) along the path to the destination.

As a result, we cannot trust fragmented UDP packets, primarily due to the low level of entropy provided by UDP port numbers and DNS message identifiers, each of which being 16 bits in size. By comparison, TCP is considered resistant against IP fragmentation attacks because TCP has a 32-bit sequence number and 32-bit acknowledgement number in each segment.
This document proposes to avoid IP fragmentation in DNS/UDP.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

"Requestor" refers to the side that sends a request. "Responder" refers to an authoritative, recursive resolver or other DNS component that responds to questions. (Quoted from EDNS0 [RFC6891])

"path MTU" is the minimum link MTU of all the links in a path between a source node and a destination node. (Quoted from [RFC8201])

Many of the specialized terms used in this document are defined in DNS Terminology [RFC8499].

3. Proposal to avoid IP fragmentation in DNS

TCP avoids fragmentation using its Maximum Segment Size (MSS) parameter, but each transmitted segment is header-size aware such that the size of the IP and TCP headers is known, as well as the far end’s MSS parameter and the interface or path MTU, in order to send a smaller segment as necessary to keep the each IP datagram below a target size. TCP’s packetizing process is also elastic as to how much queued data will fit into the next segment. DNS has no message size elasticity and lacks insight into IP header and option size, and so must make more conservative estimates about available UDP payload space.

The minimum MTU for an IPv4 interface is 576 octets, and for an IPv6 interface, 1280 octets. These are theoretic limits and no modern networks implement them. In practice, the smallest MTU witnessed in the operational DNS community is 1500 octets, the Ethernet maximum payload size. While many networks such as Packet on SONET (PoS), Fiber Distributed Data Exchange (FDDI), and Ethernet Jumbo Frame, there is no reliable way of discovering such links in an IP transmission path. Absent some kind of path MTU discovery result or a static configuration by the server or system operator, a conservative estimate must be chosen, even if less efficient.

The methods to avoid IP fragmentation in DNS are described below:

- UDP requestors and responders SHOULD send DNS responses with IP_DONTFRAG / IPV6_DONTFRAG [RFC3542] options, which will yield
either a silent timeout, or a network (ICMP) error, if the path MTU is exceeded. Upon a timeout, UDP requestors may retry using TCP or UDP, per local policy.

- The estimated maximum DNS/UDP payload size SHOULD be the actual or estimated path MTU minus the estimated header space. When actual path MTU information is not available, use the default maximum DNS/UDP payload size described in following section.

- The maximum buffer size offered by an EDNS0 requestor SHOULD be no larger than the estimated maximum DNS/UDP payload size. If the response cannot be reasonably expected fit into a buffer of that size, TCP should be used instead of UDP.

- Responders SHOULD compose UDP responses that result in IP packets that do not exceed the path MTU to the requestor. Thus, if the requestor offers a buffer size larger than estimated maximum DNS/UDP payload size, then the responder will behave as though the requestor had specified a buffer size equal to the estimated maximum DNS/UDP payload size.

- Fragmented DNS/UDP messages may be dropped before IP assembly. An ICMP error should should be sent in this case, with rate limiting to prevent this logic from becoming a DDoS amplification vector. If rate limiting is not possible, then no ICMP error should be sent. (This is a countermeasure against DNS spoofing attacks using IP fragmentation.)

The cause and effect of the TC bit is unchanged from EDNS0 [RFC6891].

4. Default maximum DNS/UDP payload size

- [RFC4035] defines that "A security-aware name server MUST support the EDNS0 message size extension, MUST support a message size of at least 1220 octets". Then, the smallest number of the maximum DNS/UDP payload size is 1220.

- However, in practice, the smallest MTU witnessed in the operational DNS community is 1500 octets. The estimated size of a DNS message’s UDP headers, IP headers, IP options, and one or more set of tunnel, IP-in-IP, VLAN, and virtual circuit headers, SHOULD be 100 octets. Then, the maximum DNS/UDP payload size may be 1400.
5. Incremental deployment

The proposed method supports incremental deployment.

When a full-service resolver implements the proposed method, the full-service resolver becomes to avoid IP fragmentation in DNS.

When an authoritative server implements the proposed method, the authoritative server becomes to avoid IP fragmentation in DNS.

6. Request to zone operator

Fat DNS responses come from fat configurations of zones. Zone operator SHOULD consider small response size configurations. For example,

- Use smaller number of name servers (13 may be too large)
- Use smaller number of A/AAAA RRs for a domain name
- Use smaller signature / public key size algorithm for DNSSEC. Signature size of ECDSA or EdDSA is smaller than RSA.

7. Considerations

In past researches ([Fujiwara2018] / dns-operations mailing list discussions), there are some authoritative servers that ignore EDNS0 requestor’s UDP payload size, and return large UDP responses.

And it is known that there are some authoritative servers that do not support TCP transport.

8. IANA Considerations

This document has no IANA actions.

9. Security Considerations

10. References

10.1. Normative References

10.2. Informative References

Appendix A. How to retrieve path MTU value to a destination

Socket options: "IP_MTU (since Linux 2.2) Retrieve the current known path MTU of the current socket. Valid only when the socket has been connected. Returns an integer. Only valid as a getsockopt(2)." (Quoted from Debian GNU Linux manual: man 7 ip)

"IPV6_MTU getsockopt(): Retrieve the current known path MTU of the current socket. Only valid when the socket has been connected. Returns an integer." (Quoted from Debian GNU Linux manual: man 7 ipv6)

Authors’ Addresses

Kazunori Fujiwara
Japan Registry Services Co., Ltd.
Chiyoda First Bldg. East 13F, 3-8-1 Nishi-Kanda
Chiyoda-ku, Tokyo 101-0065
Japan
Phone: +81 3 5215 8451
Email: fujiwara@jprs.co.jp

Paul Vixie
Farsight Security Inc
177 Bovet Road, Suite 180
San Mateo, CA 94402
Phone: +1 650 393 3994
Email: paul@redbarn.org