IPv6 Operations
Internet-Draft
Intended status: Best Current Practice
Expires: 24 August 2023

N. Elkins
M. Ackermann
INT Council
D. Dhody
India Internet Engineering Society

20 February 2023

Deep Dive into IPv6 Extension Header Testing: Behind a CDN
draft-elkins-v6ops-eh-deepdive-cdn-01

Abstract

This document proposes a methodology for isolating the location and reasons for IPv6 Extension Headers blockage in a network where the operator has access to install products and run diagnostic tests on both the client and server. The client will be outside the Content Delivery Network (CDN) and the server inside the CDN. This document will discuss the testing and topology which need to be considered when testing using a CDN infrastructure. This document is a part of the Deep Dive into EH Testing set of documents.

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 24 August 2023.

Copyright Notice

Table of Contents

1. Introduction

1.1. Problem Description

[I-D.elkins-v6ops-eh-deepdive-fw] proposes a framework to isolate the problem of where the IPv6 [RFC8200] packet is being dropped when Extension Headers (EHs) are used.

This document further proposes a framework to isolate the location and reasons for IPv6 Extension Headers blockage in a network where the operator has access to install products and run diagnostic tests on both the client and server. The client will be outside the Content Delivery Network (CDN) and the server inside the CDN.

The reason it is important to have control over both client and server is so that diagnostic tests can be run at both ends at the same time. This way, the operator can see if the packets are being sent properly and received properly.

Our initial findings are that IPv6 Extension Headers will work to the edge of the CDN. CDN providers need to be encouraged to:

* Support IPv6 to the Origin Server

* Support Extension Headers to the Origin Server

1.2. Initial Setup Requirements

The initial setup requires a:

* Client

* Server

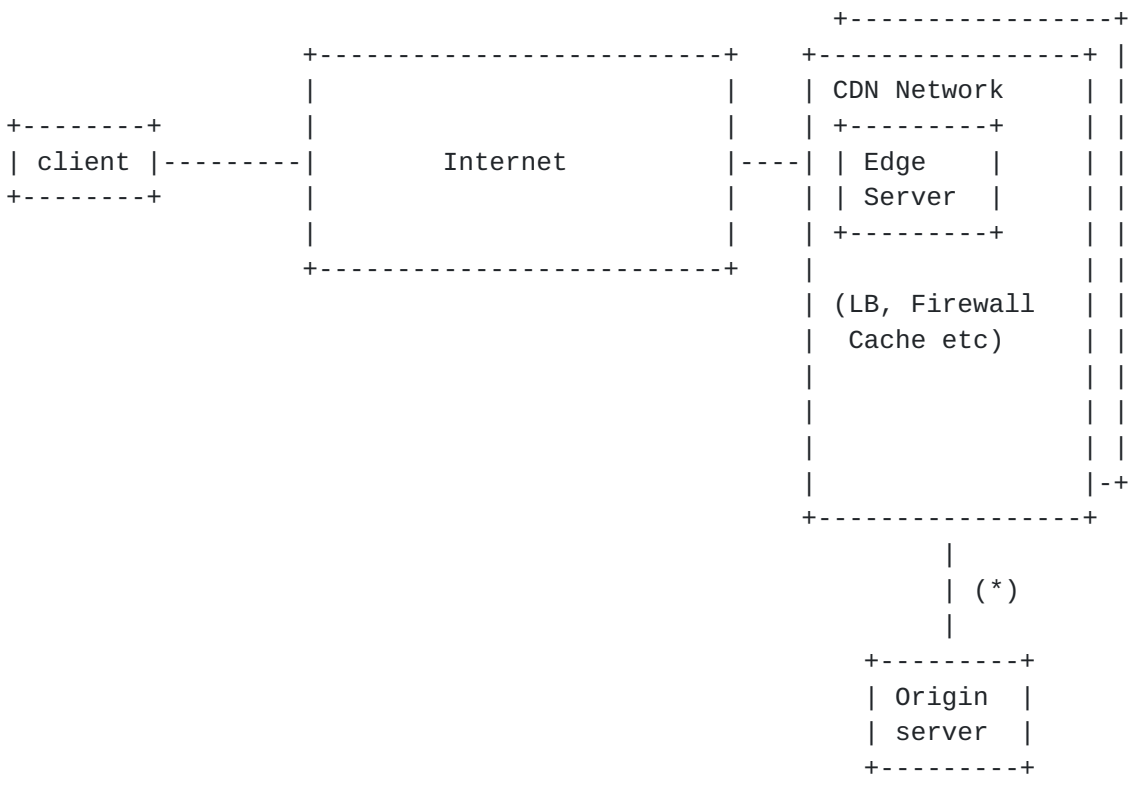
* Content Delivery Network

You also need to decide whether to craft packets with EH or to enable a client and server which have the ability to send EH along with each packet. You may wish to refer to the discussion in the [I-D.elkins-v6ops-eh-deepdive-fw] document for a detailed review of the options as well as the pros and cons of the decision. To set up the client and server, you may wish to refer to the discussion in the [I-D.elkins-v6ops-eh-deepdive-cs] document.

This document will focus on the setup and topology of the CDN network and the challenges this poses for testing.

2. CDN Topology and Concepts

You may wish to view Figure 1 for a simple topology. Clearly, within a CDN network, there can be a great deal of complexity including connections to other CDNs and so forth.



(*) - can be over the internet

Figure 1: Server behind CDN

The basic premise of a Content Delivery Network is to place servers which hold the content of a web site nearer to the client so as to speed the delivery of the desired data. Let us call these "Edge Servers". You may see the term "Cache Server" also used in some documentation on CDNs.

The web site which is the real end server, we will call the "origin server". This origin server is what we control and is enabled to send IPv6 Extension Headers.

3. Connections

In this document, we will confine ourselves to a discussion of the connections between:

* the client and the edge of the CDN

* the edge of the CDN and the origin server

These are the connections which we can observe and trace.

3.1. Connection from Client to Edge of CDN

Before initiating the connection from the client (which we control) to the CDN, we must discuss the role of DNS. When we initially place our web server (origin server) behind the CDN, we must give the CDN provider the ability to resolve our domain name, that is the CDN will become the DNS server for us.

Some CDNs use an external DNS service or resolver. In any case, what is important is that the authoritative DNS must resolve to an IP address owned by a CDN Edge server (which may vary by client location).

We must then configure the DNS to resolve the domain name into an IPv6 address or an IPv4 address. We can also tell the DNS which type of address to prefer. This will dictate the way how the connection from the client to the edge of the CDN is done.

CDN providers may differ in their support of:

* whether resolution to an IPv6 address is provided

* whether resolution to an IPv6 address is preferred

One might think that even if resolution to an IPv6 address is not preferred, we may be able to force resolution to IPv6 address by creating an IPv6-only server. This would be an incorrect assumption. We will discuss this further in the next section.

3.2. Connection from Edge of CDN to Origin Server

CDN providers may differ in their support of whether the connection from the edge of the CDN network to the origin server will be in IPv6 or IPv4. For some CDNs, it may be possible to configure IPv6 to the origin. In other cases, it is not possible to do so. That is, the connection to the origin server will travel in IPv4 regardless of whether the connection from the client to the edge of the CDN is IPv6.

As discussed previously, one might think that even if DNS resolution to an IPv6 address is not preferred, we may be able to force resolution to IPv6 address by creating an IPv6-only server. We may also apply that thinking to the connection between the edge of the CDN to the origin server. In both cases, we would be wrong. In many cases, the connection simply fails to work at all.

4. What can go wrong?

We have spent time in discussing the IPv6 support, in particular to the origin servers by CDN providers because clearly, if IPv6 itself is not supported to the origin server, then IPv6 Extension Headers to the origin server will not work.

Even in the cases where IPv6 to the origin server is provided, we know of no cases where IPv6 Extension Headers are passed from the edge of the CDN provider to the origin server. This is a WorkInProgress. We continue to work with some CDN providers to discuss this type of support.

5. Diagnostic Methodology

Once you have set up the client, server and CDN definitions properly, we may begin to test.

The following methodology assumes that the operator has:

* an Origin Server enabled to send EH with every packet

* the Origin Server is behind a CDN

* the Origin Server is running an IPv6 enabled web server (Apache / NGINX / Tomcat, et al)

* a packet trace capture tool such as TCPDump, WireShark, etc.

With this setup, only the server needs to be enabled to send EH. The client does not need to be able to send EHs, although, if that is a possibility, it will definitely add richness to the testing. You may wish to see the Deep-Dive Framework draft for more explanation on how to send EHs.

We will do two sets of tests:

1. Without EH (to test initial connectivity)

2. With EH (to test EH transmission)

5.1. Initial Test: CURL to IP Address of Server without EH

The first step is to see if the client and server can communicate properly without EH. If that is successful, we can then continue to enable EH.

Step 1: Bring up the web server on the server. Configure it to NOT send EHs Step 2: Start a TCPDump / Wireshark packet trace on the server Step 3: Start a TCPDump / Wireshark packet trace on the client Step 4: Do an HTTP CURL from the client to the Domain Name of the server.
You may wish to use HTTP rather than HTTPS to avoid problems with TLS or certificate issues.

You should see packets flowing from both ends in the packet trace. If you do not see packets, then there may be an something on the path which is stopping traffic or a configuration problem. This needs to be resolved before proceeding.

Questions you should ask:

1. Are you sending IPv6 packets from the client to the CDN Edge or Cache Server?

2. Are you sending IPv6 packets from the CDN Edge or Cache Server to the Origin Server?

If the answer to the first question is "No", then you should look at the DNS or other configuration.

If the answer to the second question is "No", then you should see if it is possible with your CDN to send IPv6 packets to the Origin Server. If this is possible, then you may have a configuration problem of some kind. This needs to be fixed before proceeding.

5.2. Second Test: CURL to IP Address of Server with EH

If you are successful with the previous tests, then you may enable EHs on the server end.

Step 1: Enable EH on the server hosting the web server Step 2: Start a TCPDump / Wireshark packet trace on the server Step 3: Start a TCPDump / Wireshark packet trace on the client Step 4: Do an HTTP CURL from the client to the Domain Name of the EH enabled test server. You may wish to use HTTP rather than HTTPS to avoid problems with TLS or certificate issues.

You should see packets flowing from both ends in the packet trace. The EHs will be from the server only. If you do not see packets, then decide which of the following problems matches what you are seeing best.

1. Packets are sent by the client but not received by the server

2. Packets are sent by the server but not received by the client

In these scenarios, you will most likely need to discuss with your CDN provider if it is possible to send IPv6 packets with EH to the Origin Server.

6. Recommendations and Further Work

Our initial findings are that IPv6 Extension Headers will work to the edge of the CDN. CDN providers need to be encouraged to:

* Support IPv6 to the Origin Server

* Support Extension Headers to the Origin Server

7. Security Considerations

This document has no security considerations.

8. Privacy Considerations

This document has no privacy considerations.

9. IANA Considerations

This document has no IANA actions.

10. References

10.1. Normative References

[RFC8200] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", STD 86, RFC 8200, DOI 10.17487/RFC8200, July 2017, <https://www.rfc-editor.org/rfc/rfc8200>.

10.2. Informative References

[I-D.elkins-v6ops-eh-deepdive-cs] Elkins, N., ackermann, M., and D. Dhody, "Deep Dive into IPv6 Extension Header Testing: Standalone Client / Server", Work in Progress, Internet-Draft, draft-elkins-v6ops-eh-deepdive-cs-00, 5 October 2022, <https://datatracker.ietf.org/doc/html/draft-elkins-v6ops-eh-deepdive-cs-00>.

[I-D.elkins-v6ops-eh-deepdive-fw]
Elkins, N., ackermann, M., and D. Dhody, "Deep Dive into IPv6 Extension Header Testing", Work in Progress, Internet-Draft, draft-elkins-v6ops-eh-deepdive-fw-01, 21 October 2022, <https://datatracker.ietf.org/doc/html/draft-elkins-v6ops-eh-deepdive-fw-01>.

Acknowledgments

TODO acknowledge.

Contributors

TODO contributors.

Authors' Addresses

Nalini Elkins
Industry Network Technology Council
United States of America
Phone: +1 831 234 4232
Email: nalini.elkins@insidethestack.com

Michael Ackermann
Industry Network Technology Council
United States of America
Phone: +1 248 703 3600
Email: mackermann@bcbsm.com
URI: https://www.bcbsm.com

Dhruv Dhody
India Internet Engineering Society
India
Email: dhruv.ietf@gmail.com