OPSAWG Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: May 7, 2020

M. Richardson
Sandelman Software Works

November 04, 2019

Operational Considerations for use of DNS in IoT devices
draft-richardson-opsawg-mud-iot-dns-considerations-01

Abstract

This document details concerns about how Internet of Things devices use IP addresses and DNS names. The issue becomes acute as network operators begin deploying RFC8520 Manufacturer Usage Description (MUD) definitions to control device access.

This document explains the problem through a series of examples of what can go wrong, and then provides some advice on how a device manufacturer can best make deal with these issues. The recommendations have an impact upon device and network protocol design.

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 May 7, 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 include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Strategies to map names 3
3. DNS and IP Anti-Patterns for IoT device Manufacturers 5
  3.1. Use of IP address literals in-protocol 5
  3.2. Use of non-deterministic DNS names in-protocol 6
  3.3. Use of a too inclusive DNS name 6
4. DNS privacy and outsourcing vs MUD controllers 7
5. Recommendations to IoT device manufacturer on MUD and DNS usage 7
  5.1. Consistently use DNS 8
  5.2. Use primary DNS names controlled by the manufacturer . . 8
  5.3. Use Content-Distribution Network with stable names . . . 8
  5.4. Prefer DNS servers learnt from DHCP/Route Advertisements 8
6. Privacy Considerations 9
7. Security Considerations 10
8. References 10
  8.1. Normative References 10
  8.2. Informative References 11
Appendix A. Appendices 12
Author's Address 12

1. Introduction

[RFC8520] provides a standardized way to describe how a specific purpose device makes use of Internet resources. Access Control Lists (ACLs) can be defined in an RFC8520 Manufacturer Usage Description (MUD) file that permit a device to access Internet resources by DNS name.

Use of a DNS name rather than IP address in the ACL has many advantages: not only does the layer of indirection permit the mapping of name to IP address to be changed over time, it also generalizes automatically to IPv4 and IPv6 addresses, as well as permitting loading balancing of traffic by many different common ways, including geography.

At the MUD policy enforcement point - the firewall - there is a problem. The firewall has only access to the layer-3 headers of the packet. This includes the source and destination IP address, and if

not encrypted by IPsec, the destination UDP or TCP port number present in the transport header. The DNS name is not present!

In order to implement this, there must be a mapping between the names in the ACLs and layer-3 IP addresses. The first section of this document details a few strategies that are used.

The second section of this document details how common manufacturer anti-patterns get in the way this mapping.

The third section of this document details how current trends in DNS resolution such as public DNS servers, DNS over TLS (DoT), and DNS over HTTPS (DoH) cause problems for the strategies employed. Poor interactions with content-distribution networks is a frequent pathology that results.

The fourth section of this document makes a series of recommendations ("best current practices") for manufacturers on how to use DNS, and IP addresses with specific purpose IoT devices.

The Privacy Considerations section concerns itself with issues that DNS-over-TLS and DNS-over-HTTPS are frequently used to deal with. The question is how these concerns apply to IoT devices located within a residence or enterprise is dealt with.

The Security Considerations section covers some of the negative outcomes should MUD/firewall managers and IoT manufacturers choose not to cooperate.

2. Strategies to map names

The simplest strategy for translating names is for a MUD controller to take is to do a DNS lookup on the name, and then use the resulting IP addresses to populate the physical ACLs.

There are a number of failures possible. The most important one is in the mapping of the names to IP addresses. [RFC1794] describes how a common mechanism that returns DNS A (or reasonably AAAA) records in a permutted order. As long as all possible A/AAAA records are returned then ACLs can be setup for all possibilities.

There are a number of circumstances in which the list is not exhaustive. The simplest is when the round robin does not return all addresses. This is routinely done by geographical DNS load balancing system. In such a system the address that is returns depends upon the network locality of the asking system. There may also be further layers of round-robin indirection.

Aside from the list of records being incomplete, the list may have changed between the time that the MUD controller did the lookup and the time that the IoT device does the lookup, and this change can result in a failure in the mapping.

In order to compensate for this, the MUD controller SHOULD regularly do DNS lookups. These lookups need to be rate limited in order to avoid load. It may be necessary to avoid recursive DNS servers in order to avoid receiving cached data. Properly designed recursive servers should cache data for many minutes to days, while the underlying DNS data can change at a higher frequency, providing different answers to different queries.

A MUD controller that is aware of which recursive DNS server that the IoT device will use can instead query that server on a periodic basis. Doing so provides three advantages:

1. any geographic load balancing will base the decision on the geolocation of the recursive DNS server, and the recursive name server will provide the same answer to the MUD controller as to the IoT device.

2. the resulting name to IP address mapping in the recursive name server will be cached, and will remain the same for the entire advertised Time-To-Live reported in the DNS query return. This also allows the MUD controller to avoid doing unnecessary queries.

3. if any addresses have been omitted in a round-robin DNS process, the cache will have the set of addresses that were returned.

The naive method of trying to map IP addresses to names will in the ACLs will not work: the reverse DNS map is frequently not populated, or if it is, it is populated with a name that is not the same name as in the MUD file ACL. This is trivial to understand when virtual hosting for web servers is considered. Many names map to a single IP address, but multiple names are seldom populated into the reverse PTR records.

Additionally, mapping IP addresses to names in real time, when making packet forwarding decisions is not practical from a performance point of view.

The solution of using the same caching recursive resolver as the target device is very simple when the MUD controllers is located in a residential CPE device. The device is usually also the policy enforcement point for the ACLs, and a caching resolver is typically located on the same device. In addition the convenience, there is a

shared fate advantage: as all three components are running on the same device, if the device is rebooted, clearing the cache, then all three components will get restarted when the device is restarted.

Where the solution is more complex is when the MUD controller is located elsewhere in an Enteprise, or remotely in a cloud such as when a Software Defines Network (SDN) is used to manage the ACLs. The DNS servers for a particular device may not be known to the MUD controller, nor the MUD controller be even permitted to make recusive queries that server if it is known. In this case, additional installation specific mechanisms are probably needed to get the right view of DNS.

3. DNS and IP Anti-Patterns for IoT device Manufacturers

This section describes a number of things with IoT manufacturers have been observed to do in the field, each of which presents difficulties for MUD enforcement points.

3.1. Use of IP address literals in-protocol

A common pattern for a number of devices is to look for firmware updates in a two step process. An initial query is made (often over HTTPS, sometimes with a POST, but the method is immaterial) to an authoritatve server. The current firmware model of the device is sometimes provided and then the authoritative server provides a determination if a new version is required, and if so, what version. In simpler cases, an HTTPS end point is queried which provides the name and URL of the most recent firmware.

The more complex case supports situations in which the device needs to be running the latest patch release before it can apply the next major release. For instance, a device running 1.4 must upgrade to at least version 1.9 before it is able to download version 2.0 of the firmware.

The authoritative upgrade server then responds with a URL of a firmware blob that the device should download and install. Best practice is that firmware is either signed internally ([I-D.ietf-suit-architecture]) so that it can be verified, or a hash of the blob is provided.

The challenge for a MUD controller is in the details of the URL that is provided. An authoritative server might be tempted to provided an IP address literal inside the protocol: there are two arguments for doing this.

One is that it eliminates problems to firmware updates that might be caused by lack of DNS, or incompatibilities with DNS. For instance bug that causes interoperability issues with some recursive servers would become unpatchable for devices that were forced to use that recursive resolver type.

A second reason to avoid a DNS in the URL is when an inhouse content-distribution system is involved that involves on-demand instances being added (or removed) from a cloud computing architecture. This model is typical of on-demand video systems including Netflix (see [LOOKING FOR NETFLIX REF], [WINDOWS UPDATE REF]), but this can occur in quite a number of other situations. Third-party content-distribution networks (CDN) tend to use DNS names in order to isolate the content-owner from changes to the distribution network.

[BEHAVE-BCP-REF] gives other good reasons why IP address literals are bad ideas; in particular they work very poorly when devices have IPv6 capabilities, and are on IPv6-only networks with NAT64 (see [RFC6146]).

3.2. Use of non-deterministic DNS names in-protocol

A second pattern is for a control protocol to connect to a known HTTP end point. This is easily described in MUD. Within that control protocol references are made to additional content at other URLs. The values of those URLs do not fit any easily described pattern and may point at arbitrary names.

Those names are often within some third-party Content-Distribution-Network (CDN) system, or may be arbitrary names in a cloud-provider storage system such as Amazon S3 (such [AmazonS3], or [Akamai]).

INSERT examples of non-deterministic CDN content.

Since it is not possible to predict a name for where the content will be, it is not possible to include that into the MUD file.

This applies to the firmware update situation as well.

3.3. Use of a too inclusive DNS name

Some CDNs make all customer content at a single URL (such as s3.amazonaws.com). This seems to be ideal from a MUD point of view: a completely predictable URL. The problem is that a compromised device could then connect to any S3 bucket, potentially attacking other buckets.

The MUD ACLs provide only for permitting end points and do not filter URLs (nor could filtering be enforced within HTTPS).

4. DNS privacy and outsourcing vs MUD controllers

[RFC7858] and [RFC8094] provide for DNS over TLS and DTLS. [I-D.ietf-dnsop-terminology-ter] details the terms. But, even with traditional DNS over Port-53 (Do53), it is possible to oursource DNS queries to other places such as the QuadX systems ([I-D.dnsop-terminology-ter] section Y).

There are significant privacy issues with having IoT devices sending their DNS queries to an outside entity. Doing it over a secure transport (DoT/DoH) is clearly better than doing so on port 53. The providers of the secure resolver service will still see the IoT device queries.

A described above in Section 2 the MUD controller needs to have access to the same resolver(s) as the IoT device. Use of the QuadX resolvers at first seems to present less of a problem than use of some other less well known resolver. While any system may use QuadX, in most cases those services are massively replicated via anycast: there is no guarantee that a MUD controller will speak to the same instance, or get the same geographic anycast result.

5. Recommendations to IoT device manufacturer on MUD and DNS usage

Inclusion of a MUD file with IoT devices is operationally quite simple. It requires only a few small changes to the DHCP client code to express the MUD URL. It can even be done without code changes via the use of a QR code affixed to the packaging (see [I-D.richardson-opsawg-securehomegateway-mud]).

The difficult part is determining what to put into the MUD file itself. There are currently tools that help with the definition and analysis of MUD files, see [mudmaker]. The remaining difficulty is now the semantic contents of what is in the MUD file. An IoT manufacturer must now spend some time reviewing what the network communications that their device does.

This document has discussed a number of challenges that occur relating to how DNS requests are made and resolved, and it is the goal of this section to make recommendations on how to modify IoT systems to work well with MUD.

5.1. Consistently use DNS

The first recommendation is to avoid using IP address literals in any protocol. Names should always be used.

5.2. Use primary DNS names controlled by the manufacturer

The second recommendation is to allocate and use names within zones controlled by the manufacturer. These names can be populated with an alias (see [RFC8499] section 2) that points to the production system. Ideally, a different name is used for each logical function, allowing for different rules in the MUD file to be enabled and disabled.

While it used to be costly to have a large number of aliases in a web server certificate, this is no longer the case. Wildcard certificates are also commonly available.

5.3. Use Content-Distribution Network with stable names

When aliases point to a Content-Distribution Network (CDN), prefer to use stable names that point to appropriately load balanced targets. CDNs that employ very low time-to-live (TTL) values for DNS make it harder for the MUD controller to get the same answer as the IoT Device. A CDN that always returns the same set of A and AAAA records, but permutes them to provide the best one first provides a more reliable answer.

5.4. Prefer DNS servers learnt from DHCP/Route Advertisements

IoT Devices should prefer doing DNS to the network provided DNS servers. Whether this is restricted to Classic DNS (Do53) or also includes using DoT/DoH is a local decision, but a locally provided DoT server SHOULD be used, as recommended by [I-D.reddy-dprive-bootstrap-dns-server] and [I-D.peterson-doh-dhcp].

Use of public QuadX resolver instead of the provided DNS resolver, whether Do53, DoT or DoH is discouraged. Should the network provide such a resolver for use, then there is no reason not to use it, as the network operator has clearly thought about this.

Some manufacturers would like to have a fallback to using a public resolver to mitigate against local misconfiguration. There are a number of reasons to avoid this, or at least do this very carefully. The recommendation here is to do this only when the provided resolvers provide no answers to any queries at all, and do so repeatedly. The use of the operator provided resolvers SHOULD be retried on a periodic basis, and once they answer, there should be no further attempts to contact public resolvers.

Finally, the list of public resolvers that might be contacted MUST be listed in the MUD file as destinations that are to be permitted. This should include the port numbers (53, 853 for DoT, 443 for DoH) that will be used as well.

6. Privacy Considerations

The use of non-local DNS servers exposes the list of names resolved to a third parties, including passive eavesdroppers.

The use of DoT and DoH eliminates the minimizes threat from passive eavesdropped, but still exposes the list to the operator of the DoT or DoH server.

The use of unencrypted (Do53) requests to a local DNS server exposes the list to any internal passive eavesdroppers, and for some situations that may be significant, particularly if unencrypted WiFi is used. Use of DoT to a local DNS recursive resolver is a preferred choice, assuming that the trust anchor for the local DNS server can be obtained, such as via [I-D.reddy-dprive-bootstrap-dns-server].

IoT devices that reach out to the manufacturer at regular intervals to check for firmware updates are informing passive eavesdroppers of the existence of a specific manufacturer's device being present at the origin location. While possession of a Large Appliance at a residence may be uninteresting, possession of intimate personal devices ("sex toys") may be a cause for embarassment.

IoT device manufacturers are encouraged to anonymizing ways to do update queries. For instance, contracting out the update notification service to a third party that deals with a large variety of devices would provide a level of defense against passive eavesdropping. Other update mechanisms should be investigated, including use of DNSSEC signed TXT records with current version information. This would permit DoT or DoH to provide the update notification. This is particularly powerful if a local recursive DoT server is used, which then communicates using DoT over the Internet.

The more complex case of section {{inprotocol} postulates that the version number needs to be provided to an intelligent agent that can decided the correct route to do upgrades. The current [I-D.ietf-suit-architecture] specification provides a wide variety of ways to accomplish the same thing without having to divulge the current version number.

The use of a publically specified firmware update protocol would also enhance privacy of IoT devices. In such a system the IoT device would never contact the manufacturer for version information or for

firmware itself. Instead, details of how to query and where to get the firmware would be provided as a MUD extension, and a Enterprise-wide mechanism would retrieve firmware, and then distribute it internally. Aside from the bandwidth savings of downloading the firmware only once, this also makes the number of devices active confidential, and provides some evidence about which devices have been upgraded and which ones might still be vulnerable. (The unpatched devices might be lurking, powered off, lost in a closet)

7. Security Considerations

This document deals with conflicting Security requirements: devices which an operator wants to manage using [RFC8520] vs requirements for the devices to get access to network resources that may be critical to their continued safe operation.

This document takes the view that the two requirements do not need to be in conflict, but resolving the conflict requires some advance planning by all parties.

8. References

8.1. Normative References

[Akamai] "Akamai", 2019, <https://en.wikipedia.org/wiki/Akamai_Technologies>.

[AmazonS3] "Amazon S3", 2019, <https://en.wikipedia.org/wiki/Amazon_S3>.

[I-D.ietf-dnsop-terminology-ter] Hoffman, P., "Terminology for DNS Transports and Location", draft-ietf-dnsop-terminology-ter-00 (work in progress), August 2019.

[I-D.ietf-suit-architecture] Moran, B., Meriac, M., Tschofenig, H., and D. Brown, "A Firmware Update Architecture for Internet of Things Devices", draft-ietf-suit-architecture-07 (work in progress), October 2019.

[I-D.peterson-doh-dhcp] Peterson, T., "DNS over HTTP resolver announcement Using DHCP or Router Advertisements", draft-peterson-doh-dhcp-01 (work in progress), October 2019.

[I-D.reddy-dprive-bootstrap-dns-server]
K, R., Wing, D., Richardson, M., and M. Boucadair, "A Bootstrapping Procedure to Discover and Authenticate DNS-over-(D)TLS and DNS-over-HTTPS Servers", draft-reddy-dprive-bootstrap-dns-server-05 (work in progress), October 2019.

[RFC1794] Brisco, T., "DNS Support for Load Balancing", RFC 1794, DOI 10.17487/RFC1794, April 1995, <https://www.rfc-editor.org/info/rfc1794>.

[RFC6146] Bagnulo, M., Matthews, P., and I. van Beijnum, "Stateful NAT64: Network Address and Protocol Translation from IPv6 Clients to IPv4 Servers", RFC 6146, DOI 10.17487/RFC6146, April 2011, <https://www.rfc-editor.org/info/rfc6146>.

[RFC7858] Hu, Z., Zhu, L., Heidemann, J., Mankin, A., Wessels, D., and P. Hoffman, "Specification for DNS over Transport Layer Security (TLS)", RFC 7858, DOI 10.17487/RFC7858, May 2016, <https://www.rfc-editor.org/info/rfc7858>.

[RFC8094] Reddy, T., Wing, D., and P. Patil, "DNS over Datagram Transport Layer Security (DTLS)", RFC 8094, DOI 10.17487/RFC8094, February 2017, <https://www.rfc-editor.org/info/rfc8094>.

[RFC8499] Hoffman, P., Sullivan, A., and K. Fujiwara, "DNS Terminology", BCP 219, RFC 8499, DOI 10.17487/RFC8499, January 2019, <https://www.rfc-editor.org/info/rfc8499>.

[RFC8520] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", RFC 8520, DOI 10.17487/RFC8520, March 2019, <https://www.rfc-editor.org/info/rfc8520>.

8.2. Informative References

[I-D.richardson-opsawg-securehomegateway-mud]
Richardson, M., Latour, J., and F. Khan, "MUD processing and extensions for Secure Home Gateway Project", draft-richardson-opsawg-securehomegateway-mud-01 (work in progress), March 2019.

[mudmaker]
"Mud Maker", 2019, <https://mudmaker.org>.

Appendix A. Appendices

Author's Address

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca