Workgroup: OPSAWG Working Group
Internet-Draft:
draft-ietf-opsawg-mud-iot-dns-considerations-08
Published: 24 January 2023
Intended Status: Best Current Practice
Expires: 28 July 2023
Authors: M. Richardson
Sandelman Software Works
W. Pan
Huawei Technologies

Operational Considerations for use of DNS in IoT devices

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 makes recommendations on when and how to use DNS names in MUD files.

About This Document

This note is to be removed before publishing as an RFC.

Status information for this document may be found at https://datatracker.ietf.org/doc/draft-ietf-opsawg-mud-iot-dns-considerations/.

Discussion of this document takes place on the opsawg Working Group mailing list (mailto:opsawg@ietf.org), which is archived at https://mailarchive.ietf.org/arch/browse/opsawg/.

Source for this draft and an issue tracker can be found at https://github.com/mcr/iot-mud-dns-considerations.

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 28 July 2023.

Table of Contents

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 multi-CDN deployments wherein load balancing can account for geography and load.

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!

It has been suggested that one answer to this problem is to provide a forced intermediate for the TLS connections. This could in theory be done for TLS 1.2 connections. The MUD policy enforcement point could observe the Server Name Identifier (SNI) [RFC6066]. Some Enterprises do this already. But, as this involves active termination of the TCP connection (a forced circuit proxy) in order to see enough of the traffic, it requires significant effort. In TLS 1.3 with or without the use of ECH, middlebox cannot rely on SNI inspection because a malware could lie about the SNI and middlebox without acting as a TLS proxy does not have visibility into the server certificate.

So in order to implement these name based ACLs, 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 presolution such as public DNS servers, DNS over TLS (DoT), DNS over QUIC (DoQ), and DNS over HTTPS (DoH) cause problems for the strategies employed. Poor interactions with content-distribution networks is a frequent pathology that can result.

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. How these concerns apply to IoT devices located within a residence or enterprise is a key concern.

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

2. Terminology

Although this document is not an IETF Standards Track publication, it adopts the conventions for normative language to provide clarity of instructions to the implementer. 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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

3. Strategies to map names

The most naive method is to try to map IP addresses to names using the in-addr.arpa (IPv4), and ipv6.arpa (IPv6) mappings.

3.1. Failing strategy

Attempts to map IP address to names in real time fails for a number of reasons:

1. it can not be done fast enough,

2. it reveals usage patterns of the devices,

3. the mapping are often incomplete,

4. even if the mapping is present, due to virtual hosting, it may not map back to the name used in the ACL.

This is not a successful strategy, its use is NOT RECOMMENDED for the reasons explained below.

3.1.1. Too slow

Mapping of IP address to names requires a DNS lookup in the in-addr.arpa or ip6.arpa space. For a cold DNS cache, this will

typically require 2 to 3 NS record lookups to locate the DNS server that holds the information required. At 20 to 100ms per round trip, this easily ads up to significant time before the packet that caused the lookup can be released.

While subsequent connections to the same site (and subsequent packets in the same flow) will not be affected if the results are cached, the effects will be felt. The ACL results can be cached for a period of time given by the TTL of the DNS results, but the lookup must be performed again in a number of hours to days.

3.1.2. Reveals patterns of usage

By doing the DNS lookups when the traffic occurs, then a passive attacker can see when the device is active, and may be able to derive usage patterns. They could determine when a home was occupied or not. This does not require access to all on-path data, just to the DNS requests to the bottom level of the DNS tree.

3.1.3. Mappings are often incomplete

A service provider that fails to include an A or AAAA record as part of their forward name publication will find that the new server is simply not used. The operational feedback for that mistake is immediate. The same is not true for reverse names: they can often be incomplete or incorrect for months or even years without visible affect on operations.

Service providers often find it difficult to update reverse maps in a timely fashion, assuming that they can do it at all. Many cloud based solutions dynamically assign IP addresses to services, often as the service grows and shrinks, reassigning those IP addresses to other services quickly. The use of HTTP 1.1 Virtual Hosting may allow addresses and entire front-end systems to be re-used dynamically without even reassigning the IP addresses.

In some cases there are multiple layers of CNAME between the original name and the target service name. This is often due to a layer of load balancing in DNS, followed by a layer of load balancer at the HTTP level.

The reverse name for the IP address of the load balancer usually does not change. If hundreds of web services are funnelled through the load balancer, it would require hundreds of PTR records to be deployed. This would easily exceed the UDP/DNS and EDNS0 limits, and require all queries to use TCP, which would further slow down loading of the records.

The enumeration of all services/sites that have been at that load balancer might also consistitute a security concern. To liimt churn

of DNS PTR records, and reduce failures of the MUD ACLs, operators would want to add all possible names for each reverse name, whether or not the DNS load balancing in the forward DNS space lists that end-point at that moment.

3.1.4. Names can have wildcards

In some large hosting providers content is hosted under some URL that includes a wildcard. For instance, github.io, which is used for hosted content, including the Editors' copy of internet drafts stored on github, does not actually publish any names. Instead a wildcard exists to answer.

github would be unable to provision all infinity of possible names into the PTR records.

3.2. A successful strategy

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

There are still a number of failures possible.

The most important one is in the mapping of the names to IP addresses may be non-deterministic. [RFC1794] describes the very common mechanism that returns DNS A (or reasonably AAAA) records in a permutted order. This is known as Round Robin DNS, and it has been used for many decades. The device is intended to use the first IP address that is returned, and each query returns addresses in a different ordering, splitting the load among many servers.

This situation does not result in failures as long as all possible A/AAAA records are returned. The MUD controller and the device get a matching set, and the ACLs that are setup cover 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. It can also happen if there are more addresses than will conveniently fit into a DNS reply. The reply will be marked as truncated. (If DNSSEC resolution will be done, then the entire RR must be retrieved over TCP (or using a larger EDNS(0) size) before being validated)

However, in a geographical DNS load balancing system, different answers are given based upon the locality of the system asking. 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 for the ACL to match.

In order to compensate for this, the MUD controller SHOULD regularly do DNS lookups in order to get never have stale data. These lookups need to be rate limited in order to avoid load. It may be necessary to avoid local recursive DNS servers. The MUD controller SHOULD incorporate its own recursive caching DNS server. 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 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.

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

In many design fields, there are good patterns that should be emulated, and often there are patterns that should not be emulated. The latter are called anti-patterns, as per [antipatterns].

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.

4.1. 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 a vendor system that knows whether or not an update is required.

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 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 ([RFC9019]) so that it can be verified, or a hash of the blob is provided.

An authoritative server might be tempted to provided an IP address literal inside the protocol: there are two arguments (anti-patterns) 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 the 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.

But, there are many problems with use of IP address literals for the location of the firmware.

The first is that the update service server must decide whether to provide an IPv4 or an IPv6 literal. A DNS name can contain both kinds of addresses, and can also contain many different IP addresses of each kind.

The second problem is that it forces the MUD file definition to contain the exact same IP address literals. It must also contain an ACL for each address literal. DNS provides a useful indirection method that naturally aggregates the addresses.

A third problem involves the use of HTTPS. IP address literals do not provide enough context for TLS ServerNameIndicator to be useful [RFC6066]. This limits the firmware repository to be a single tenant on that IP address, and for IPv4 (at least), this is no longer a sustainable use of IP addresses.

And with any non-determistic name or address that is returned, the MUD controller is not challenged to validate the transaction, as it can not see into the communication.

Third-party content-distribution networks (CDN) tend to use DNS names in order to isolate the content-owner from changes to the distribution network.

4.2. 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]).

Such names may be unpredictably chosen by the content provider, and not the content owner, and so impossible to insert into a MUD file.

Even if the content provider chosen names are deterministic they may change at a rate much faster than MUD files can be updated.

This in particular may apply to the location where firmware updates may be retrieved.

4.3. 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 the contents of any bucket, potentially attacking the data from other customers.

Exactly what the risk is depends upon what the other customers are doing: it could be limited to simply causing a distributed denial of service attack resulting to high costs to those customers, or such an attack could potentially include writing content.

Amazon has recognized the problems associated with this practice, and aims to change it to a virtual hosting model, as per [awss3virtualhosting].

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

5. DNS privacy and outsourcing versus MUD controllers

[RFC7858] and [RFC8094] provide for DNS over TLS (DoT) and DNS over HTTPS (DoH). [I-D.ietf-dnsop-terminology-ter] details the terms. But, even with traditional DNS over Port-53 (Do53), it is possible to outsource DNS queries to other public services, such as those operated by Google, CloudFlare, Verisign, etc.

For some users and classes of device, revealing the DNS queries to those outside entities may consititute a privacy concern. For other users the use of an insecure local resolver may constitute a privacy concern.

A described above in Section 3 the MUD controller needs to have access to the same resolver(s) as the IoT device.

6. 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 [RFC9238].

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.

6.1. Consistently use DNS

For the reasons explained in Section 4.1, the most important recommendation is to avoid using IP address literals in any protocol. Names should always be used.

6.2. 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 which allowed for an infinite number of possible names.

6.3. 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.

6.4. Do not use geofenced names

Due the problems with different answers from different DNS servers, described above, a strong recommendation is to avoid using such things.

6.5. 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-add-iot-byod-bootstrap].

The ADD WG is currently only focusing on insecure discovery mechanisms like DHCP/RA [I-D.ietf-add-dnr] and DNS based discovery mechanisms ([I-D.ietf-add-ddr]).

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.

It is recommended that use of non-local resolvers is only done when the locally 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.

7. 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. There are additional methods, such as described by [I-D.pauly-dprive-oblivious-doh].

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 Encrypted DNS connection 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-add-iot-byod-bootstrap].

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.

Identifying the IoT device type empowers the attacker to launch targeted attacks to the IoT device (e.g., Attacker can advantage of the device vulnerability).

While possession of a Large (Kitchen) Appliance at a residence may be uninteresting to most, possession of intimate personal devices (e.g., "sex toys") may be a cause for embarrassment.

IoT device manufacturers are encouraged to find ways to anonymize their 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 convey the update notification in a private fashion. 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 Section 4.1 postulates that the version number needs to be provided to an intelligent agent that can decided the correct route to do upgrades. The current [RFC9019] 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)

8. Security Considerations

This document deals with conflicting Security requirements:

1. devices which an operator wants to manage using [RFC8520]

2. 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.

9. References

9.1. 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. E., "Terminology for DNS Transports and Location", Work in Progress, Internet-Draft, draft-ietf-dnsop-terminology-ter-02, 3 August 2020, <https://www.ietf.org/archive/id/draft-ietf-dnsop-terminology-ter-02.txt>.

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

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[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>.

[RFC9019] Moran, B., Tschofenig, H., Brown, D., and M. Meriac, "A Firmware Update Architecture for Internet of Things", RFC 9019, DOI 10.17487/RFC9019, April 2021, <https://www.rfc-editor.org/info/rfc9019>.

9.2. Informative References

[antipatterns] "AntiPattern", 12 July 2021, <https://www.agilealliance.org/glossary/antipattern>.

[awss3virtualhosting] "Down to the Wire: AWS Delays 'Path-Style' S3 Deprecation at Last Minute", 12 July 2021, <https://techmonitor.ai/techonology/cloud/aws-s3-path-deprecation>.

[I-D.ietf-add-ddr] Pauly, T., Kinnear, E., Wood, C. A., McManus, P., and T. Jensen, "Discovery of Designated Resolvers", Work in Progress, Internet-Draft, draft-ietf-add-ddr-10, 5 August 2022, <https://www.ietf.org/archive/id/draft-ietf-add-ddr-10.txt>.

[I-D.ietf-add-dnr] Boucadair, M., Reddy.K, T., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for the Discovery of Network-designated Resolvers (DNR)", Work in Progress, Internet-Draft, draft-ietf-add-dnr-13, 13 August 2022, <https://www.ietf.org/archive/id/draft-ietf-add-dnr-13.txt>.

[I-D.pauly-dprive-oblivious-doh] Kinnear, E., McManus, P., Pauly, T., Verma, T., and C. A. Wood, "Oblivious DNS over HTTPS", Work in Progress, Internet-Draft, draft-pauly-dprive-oblivious-doh-11, 17 February 2022, <https://www.ietf.org/archive/id/draft-pauly-dprive-oblivious-doh-11.txt>.

[I-D.peterson-doh-dhcp] Peterson, T., "DNS over HTTP resolver announcement Using DHCP or Router Advertisements", Work in Progress, Internet-Draft, draft-peterson-doh-dhcp-01, 21 October 2019, <https://www.ietf.org/archive/id/draft-peterson-doh-dhcp-01.txt>.

[I-D.reddy-add-iot-byod-bootstrap] Reddy.K, T., Wing, D., Richardson, M., and M. Boucadair, "A Bootstrapping Procedure to Discover and Authenticate DNS-over-TLS and DNS-over-HTTPS Servers for IoT and BYOD Devices", Work in Progress, Internet-Draft, draft-reddy-add-iot-byod-bootstrap-01, 26 July 2020, <https://www.ietf.org/archive/id/draft-reddy-add-iot-byod-bootstrap-01.txt>.

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

[RFC6066] Eastlake 3rd, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, DOI 10.17487/RFC6066, January 2011, <https://www.rfc-editor.org/info/rfc6066>.

[RFC9238]
Richardson, M., Latour, J., and H. Habibi Gharakheili, "Loading Manufacturer Usage Description (MUD) URLs from QR Codes", RFC 9238, DOI 10.17487/RFC9238, May 2022, <https://www.rfc-editor.org/info/rfc9238>.

Appendix A. Appendices

Contributors

Tirumaleswar Reddy
Nokia

Authors' Addresses

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com