OPSAWG Working Group
Internet-Draft
Intended status: Best Current Practice
Expires: 12 January 2022

M. Richardson
Sandelman Software Works
W. Pan
Huawei Technologies
11 July 2021

Operational Considerations for use of DNS in IoT devices
draft-ietf-opsawg-mud-iot-dns-considerations-02

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.

{RFC-EDITOR, please remove. Markdown and issue tracker for this document is at https://github.com/mcr/iot-mud-dns-considerations.git }

Commented [DT1]: I don't think this is correct, MUD firewalls can be deployed by enterprises, homeowners, etc., not just entities people normally think of as network operators. Suggest "network administrators"

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 12 January 2022.

Copyright Notice

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.

Commented [DT2]: But not by IP literal? If so, that would seem to be a flaw in MUD, since you couldn't whitelist your DNS server address for example. The last paragraph of section 6.5 below would seem to imply you can, i.e. that this sentence is incorrect.

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 has 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~~ intermediary 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 it requires significant effort, 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~~. But, TLS 1.3 provides options to encrypt the SNI as the ESNI, which renders the practice useless in the end.

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 of this mapping.

The third section of this document details how current trends in DNS ~~p~~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 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.

Commented [DT3]: This isn't correct, it also has access to the layer-2 headers. And if ipsec isn't used then it does have access to layer 4 headers.

Commented [DT4]: Undefined term

Commented [DT5]: But so does this third section (SHOULD appears in section 3) so this is not really a summary unique to the fourth section.

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

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document is a Best Current Practices (BCP) document. It uses the above language where it needs to make a normative requirement ~~on~~for implementations.

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. This 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, and do not use it.

XXX -- explain in detail how this can fail.

XXX -- explain N:1 vs 1:1 for virtual hosting.

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) on the name in the MUD file, 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 permu~~t~~ted order. This is known as Round Robin DNS, and it has been

Commented [DT6]: Clarify: fast enough for what? For example, if an IoT device only sends 1 packet every 10 seconds, why would it not be fast enough?

Commented [DT7]: Reveals to whom? Elaborate.

Commented [DT8]: Be more specific. I think you mean that reverse mappings are often not present at all, not that that're present but "incomplete" somehow.

Commented [DT9]: Use 2119 language?

Commented [DT10]: These need to be filled in of course, after which this document should go through an iot review again.

Commented [DT11]: RFC 8520 says this is "a synonym that has been used *in the past* for MUD manager " To my reading, that means it should not be used in any new documents, only "MUD manager"

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 set_up cover all possibilities.

There are a number of circumstances however in which the list is not exhaustive. The simplest is when the ~~round robin~~DNS lookup does not return all addresses. This is routinely done by geographical DNS load balancing systems. 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~~ does the lookup and the time that the IoT device does the lookup, and this change can result in a failure ~~for~~ of the ACL to match.

~~In order t~~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.

Commented [DT12]: Two words when used as a verb (setup is a noun)

Commented [DT13]: In which case it can't really be called round robin per se, hence reword for clarity

Commented [DT14]: Not sure what the relevance of this statement is. Is this a recommendation to IoT devices or what? Recommend removing.

Commented [DT15]: The resolutions can happen in either order (e.g., if you reboot the MUD controller) so use "does" for both.

Commented [DT16]: Load where? At the MUD enforcement point? Is the "need to be" a recommendation that is supposed to be a SHOULD?

Commented [DT17]: A MUD manager MAY?

Commented [DT18]: Is this specified in the MUD file? If not, how would it know?

Commented [DT19]: If it doesn't query the same server the IoT device uses then it doesn't solve the problem since one or the other may be serving content cached at a different time and hence have newer or older addresses, that result in a failure. I would recommend explicitly stating that MUD files for IoT devices must specify what DNS server is used so the MUD controller can use the same one.

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 to 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 eEnterprise, or remotely in a cloud such as when a Software Defineds Network (SDN) is used to manage the ACLs. The DNS servers for a particular IoT device may not be known to the MUD controller, nor might the MUD controller be even be permitted to make recursive queries to 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 that 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 an authoritative server. (What is this) 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 ([I-D.ietf-suit-architecture]) so that it can be verified, or a hash of the blob is provided.

Commented [DT20]: This is only true if the target device uses the one provided by the network, and not its own. But it's also simple if the MUD file tells the MUD controller what to use.

Commented [DT21]: Confusing terminology since "target device" was used earlier in this paragraph to mean the IoT device I think, whereas here "the device" is referring to the MUD manager I think. Replace all uses of "device" with either "IoT device" or "MUD manager" for clarity.

Commented [DT22]: Firmware updates is not the only scenario where there could be IP literals in-protocol. If this is just an example, say so. Section 4.2 is much better since it is worded generically and only uses firmware update as an example

Commented [DT23]: This looks like a TODO

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.

T~~But, t~~here are, however, many problems with the use of IP address literals ~~for the location of the firmware~~.

The first is that the ~~update service~~ server provider 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~~server 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-deterministic 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.

Commented [DT24]: Arguments != anti-patterns. Anti-patterns are designs/implementations, not the reasons for doing them.

Commented [DT25]: Again this isn't just about firmware updates, the same text here can apply any time the IoT device talks to a well-known server (especially a manufacturer server) for any type of functionality, including for doing whatever the IoT device is supposed to do in normal operation.

Commented [DT26]: Not true. They can choose to provide a list of any number of each. They need not decide on only one, just like they decide what entries to put in DNS for the name.

Commented [DT27]: Logic here does not hold, the same list could be provided by other means, whether in the firmware image, or in some other protocol (like for doing whatever the IoT device was built to do). So in reality this is not necessarily a problem with the use of IP address literals, it's a problem with only providng one address and that same problem would occur if they only populated one address in the DNS name.

Commented [DT28]: This is true, but this doesn't explain why you view it as a problem. The devil's advocate would say this is not a problem, it's a feature. The problem is, as stated earlier, they would argue, the lack of dns or incompatibilities with DNS, etc.

Commented [DT29]: If someone is already doing this, are you telling them that they're unsustainable? Or only that new entrants can't do this? Or that they have to use a multi-tenant cloud provider instead of self-hosting a server? I can't tell what you mean by sustainable here. Either way, using IPv4 as a justification here seems suspect since it seems for IoT devices the recommendation should be to use IPv6 in which case IPv4 problems are better served by using IPv6 than by (say) telling them to not use IP literals per se. Not that I'm a fan of IP literals personally, but I think for a BCP the case here is not made.

Commented [DT30]: Shouldn't this say "enforcement point"? The manager/controller is not necessarily in the path of the transaction, only the enforcement point is.

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]~~)~~.

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.

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 any S3 bucket, potentially attacking other buckets.

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.

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, however, still see the IoT device queries.

A described above in Section 3 the MUD controller needs to have access to the same resolver(s) as the IoT device. Use of the QuadX resolvers (such as Google's 8.8.8.8) 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.

Commented [DT31]: Can't parse grammar. See suggested fix.

Commented [DT32]: Undefined term, what's this?

Commented [DT33]: I can't tell what this means.

Commented [DT34]: But I think such issues do not apply when the IoT manufacturer is the provider of the secure resolver service.

Commented [DT35]: Undefined term

XXX - THIS NEEDS WAY MORE EXPLANATION.

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 [I-D.richardson-mud-qrcode].

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

The first 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 allow~~ed~~ for an infinite number of possible names.

Commented [DT36]: Is there anything in this document that implies one SHOULD have a MUD file? Or only that IF you have a MUD file THEN the recommendations in this BCP apply?

Commented [DT37]: This doesn't match section 1.5 of RFC 8520 which says there are also X.509 and LLDP alternatives.

Commented [DT38]: I don't think this recommendation is yet justified in this document. (See also my comments in section 4.1.) But more specifically: why "IP address literals" only in string form? Are they ok if in binary form? E.g.,if my IoT protocol uses CBOR is it ok to pass IP address blobs? (Seems like there should be no difference in recommendation between strings vs binary.) And I will observe that DNS itself passes IP addresses so the "any protocol" part needs thought when addressing the IP-addresses-in-binary-form part.

Commented [DT39]: What do you mean by "primary" here? RFC 8499 has "primary master" and "primary server" but never "primary name"

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

XXX - it has been suggested that this will not help, thus previous recommendation.

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].

The ADD WG is currently only focusing on insecure discovery mechanisms like DHCP/RA [I-D.btw-add-home] and DNS based discovery mechanisms ({?{I-D.pauly-add-deer}}). Secure discovery of network provided DoH/DoT resolver is possible using the mechanisms discussed in [I-D.reddy-add-enterprise] section-4.

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 IoT device 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.

Commented [DT40]: Undefined term. Do you mean the same as what was earlier referred to as a "geographical DNS load balancing system"?

Commented [DT41]: Unclear whether this note means this section will go away or be rewritten or stand as is

Commented [DT42]: SHOULD?

Commented [DT43]: This document does not provide sufficient justification yet for this recommendation. Specifically, it does not state what the problem is if the IoT manufacturer provides its own DNS resolver service and puts it in the MUD file so the MUD manager knows about it. (I could imagine an argument saying that's not a "current practice" so may or may not be better or worse but in a BCP one has to have current practice, but this document currently doesn't make such an argument.)

Commented [DT44]: "NOT RECOMMENDED"?

Commented [DT45]: I think this sentence is not true. You have not proven the non-existence of a reason. Network operator goals and IoT device manufacturer goals may not be aligned. Just because a network operator has clearly thought about something does not mean that all IoT manufacturers would agree with their conclusion. Remove this claim or provide a non-existence proof. (And in my opinion the first sentence of the next paragraph is a reason they would cite, and the doc even admits it may be valid enough to "do very carefully".)

Commented [DT46]: Use 2119 language

Commented [DT47]: SHOULD NOT be?

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 eavesdroppe~~red~~, 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-dprive-bootstrap-dns-server].

IoT devices that directly 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~~ against the IoT device (e.g., an a~~A~~ttacker ~~can~~ might take advantage of ~~the~~ a known 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.

Commented [DT48]: As phrased, this sentence is not true. It is only true if the non-local DNS server is actually operated by a third party, or if communication to it is not encrypted. If for example you do DoT/DoH to a DNS server hosted by the IoT manufacturer, it's non-local but does not expose the list of names to any third party as far as I can tell.

Commented [DT49]: True, but if that operator is the IOT manufacturer it's not a third party. The network operator on the other hand could be considered a third party (the IOT device owner and the IOT manufacturer being the first two parties) so using the operator provided DNS could even be argued to be worse from a privacy POV. They can see the IP addresses for sure, but if there is any addition PII in the names, then using the operator provided DNS leaks the names to an additional third party that a manufacturer provided DNS server does not.

Commented [DT50]: Undefined (in this document) term, cite a doc with a definition of it such as RFC 4949 for example

Commented [DT51]: This issue isn't just about firmware updates

Commented [DT52]: A more common example typically used is privacy concerns with health care devices (insulin machines, blood pressure monitors, etc) given HIPAA etc laws in various jurisdictions. It's not just embarrassment, it may be arguably illegal depending on the jurisdiction.

Commented [DT53]: SHOULD? MAY?

Commented [DT54]: SHOULD?

Commented [DT55]: By each IoT manufacturer? By the IETF? "Should investigate" doesn't sound like a best current practice for IoT device manufacturers, so I would argument it does not belong in this section.

The more complex case of ~~section~~ Section 4.1 postulates that the version number needs to be provided to an intelligent agent that can decide~~d~~ 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 public~~al~~ly 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, or even 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., "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>.

Commented [DT56]: Disagree with this as worded. The benefit is any time a service can be used that isn't IoT manufacturer specific, you get the benefit. So by analogy, Windows Update provides Windows drivers from many manufacturers without having to go to the manufacturer itself, it's the use of a common service, NOT whether such a service uses a publicly specified protocol per se. Whether having a publicly specified update protocol would or would not encourage more common update services I don't know, but SUIT for example was not chartered to do a protocol, in part because it wasn't clear anyone wanted a public protocol given the prevalence of proprietary ones now and the apparent lack of any requirement for interoperability other than the manifest format that SUIT is specifying (not a protocol).

Commented [DT57]: This document isn't just for enterprises.

Commented [DT58]: Confidential from whom?

Commented [DT59]: I that this document deals with any conflict cases. A conflict would arise only if the operator wanted a device to NOT get access to network resources that may be critical to their continued safe operation. Right now it only discusses cases where these two requirements are aligned, i.e., everything in the manufacturer's MUD file is permitted. Suggest rewording here and the last sentence of this section to not imply there is any conflict to resolve.

Commented [DT60]: I don't like this wording, as it implies MUD cannot be used by consumer-purchased CPEs for IOT devices in their home. Use of MUD should permit, but not require there to be someone who actively wants to "manage" devices. Perhaps "a network administrator wants to ensure that IoT devices only access approved resources"

[I-D.ietf-suit-architecture]
Moran, B., Tschofenig, H., Brown, D., and M. Meriac, "A Firmware Update Architecture for Internet of Things", Work in Progress, Internet-Draft, draft-ietf-suit-architecture-16, 27 January 2021, <https://www.ietf.org/archive/id/draft-ietf-suit-architecture-16.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-dprive-bootstrap-dns-server]
Reddy, T., Wing, D., Richardson, M. C., and M. Boucadair, "A Bootstrapping Procedure to Discover and Authenticate DNS-over-TLS and DNS-over-HTTPS Servers", Work in Progress, Internet-Draft, draft-reddy-dprive-bootstrap-dns-server-08, 6 March 2020, <https://www.ietf.org/archive/id/draft-reddy-dprive-bootstrap-dns-server-08.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>.

[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>.

[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>.

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.btw-add-home] Boucadair, M., Reddy, T., Wing, D., Cook, N., and T. Jensen, "DHCP and Router Advertisement Options for Encrypted DNS Discovery", Work in Progress, Internet-Draft, draft-btw-add-home-12, 22 January 2021, <https://www.ietf.org/archive/id/draft-btw-add-home-12.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-06, 8 March 2021, <https://www.ietf.org/archive/id/draft-pauly-dprive-oblivious-doh-06.txt>.

[I-D.reddy-add-enterprise] Reddy, T. and D. Wing, "DNS-over-HTTPS and DNS-over-TLS Server Deployment Considerations for Enterprise Networks", Work in Progress, Internet-Draft, draft-reddy-add-enterprise-00, 23 June 2020, <https://www.ietf.org/archive/id/draft-reddy-add-enterprise-00.txt>.

[I-D.richardson-mud-qrcode]
Richardson, M., Latour, J., and H. H. Gharakheili, "On loading MUD URLs from QR codes", Work in Progress, Internet-Draft, draft-richardson-mud-qrcode-00, 17 December 2020, <https://www.ietf.org/archive/id/draft-richardson-mud-qrcode-00.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>.

Appendix A. Appendices

Authors' Addresses

Michael Richardson
Sandelman Software Works

Email: mcr+ietf@sandelman.ca

Wei Pan
Huawei Technologies

Email: william.panwei@huawei.com