Internet Engineering Task Force (IETF)
Request for Comments: 9039
Category: Standards Track
ISSN: 2070-1721

J. Arkko
Ericsson
C. Jennings
Cisco
Z. Shelby
Edge Impulse

June 2021

Uniform Resource Names for Device Identifiers

Abstract

This document describes a new Uniform Resource Name (URN) namespace for hardware device identifiers. A general representation of device identity can be useful in many applications, such as in sensor data streams and storage or in equipment inventories. A URN-based representation can be passed along in applications that need the information.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841.

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9039.

Copyright Notice

Table of Contents

1. Introduction

This document describes a new Uniform Resource Name (URN) [RFC8141] namespace for hardware device identifiers. A general representation of device identity can be useful in many applications, such as in sensor data streams and storage or in equipment inventories [RFC7252] [RFC8428] [CoRE-RD].

A URN-based representation can be passed along in applications that need the information. It fits particularly well for protocols mechanisms that are designed to carry URNs [RFC7230] [RFC7540] [RFC3261] [RFC7252]. Finally, URNs can also be easily carried and stored in formats such as XML [W3C.REC-xml-19980210], JSON [RFC8259], or SenML [RFC8428]. Using URNs in these formats is often preferable as they are universally recognized and self-describing and therefore avoid the need to agree to interpret an octet string as a specific form of a Media Access Control (MAC) address, for instance. Passing URNs may consume additional bytes compared to, for instance, passing 4-byte binary IPv4 addresses, but the former offers some flexibility in return.

This document defines identifier URN types for situations where no such convenient type already exists. For instance, [RFC6920] defines cryptographic identifiers, [RFC7254] defines International Mobile station Equipment Identity (IMEI) identifiers for use with 3GPP cellular systems, and [RFC8464] defines Mobile Equipment Identity (MEID) identifiers for use with 3GPP2 cellular systems. Those URN types should be employed when such identifiers are transported; this document does not redefine these identifiers in any way.

Universally Unique Identifier (UUID) URNs [RFC4122] are another alternative way to represent device identifiers and already support MAC addresses as one type of identifier. However, UUIDs can be inconvenient in environments where it is important that the identifiers be as simple as possible and where additional requirements on stable storage, real-time clocks, and identifier length can be prohibitive. Often, UUID-based identifiers are preferred for general purpose uses instead of the MAC-based device URNs defined in this document. The device URNs are recommended for constrained environments.

Future device identifier types can extend the device URN type defined in this document (see Section 7), or they can define their own URNs.

Note that long-term stable unique identifiers are problematic for privacy reasons and should be used with care as described in [RFC7721].

The rest of this document is organized as follows. Section 3 defines the "DEV" URN type, and Section 4 defines subtypes for IEEE MAC-48, EUI-48 and EUI-64 addresses, and 1-Wire device identifiers. Section 5 gives examples. Section 6 discusses the security and privacy considerations of the new URN type. Finally, Section 7 specifies the IANA registration for the new URN type and sets requirements for subtype allocations within this type.

2. Requirements Language

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. DEV URN Definition

Namespace Identifier: "dev"

Version: 1

Date: 2020-06-24

Registrant: IETF and the CORE Working Group. Should the working group cease to exist, discussion should be directed to the Applications and Real-Time Area or general IETF discussion forums, or the IESG.

3.1. Purpose

The DEV URNs identify devices with device-specific identifiers such as network card hardware addresses. DEV URNs are scoped to be globally applicable (see [RFC8141], Section 6.4.1) and, in general, enable systems to use these identifiers from multiple sources in an interoperable manner. Note that in some deployments, ensuring uniqueness requires care if manual or local assignment mechanisms are used, as discussed in Section 3.3.

Some typical DEV URN applications include equipment inventories and smart object systems.

DEV URNs can be used in various ways in applications, software systems, and network components, in tasks ranging from discovery (for instance, when discovering 1-Wire network devices or detecting MAC-addressable devices on a LAN) to intrusion detection systems and simple catalogues of system information.

While it is possible to implement resolution systems for specific applications or network locations, DEV URNs are typically not used in a way that requires resolution beyond direct observation of the relevant identifier fields in local link communication. However, it is often useful to be able to pass device identifier information in generic URN fields in databases or protocol fields, which makes the use of URNs for this purpose convenient.

The DEV URN namespace complements existing namespaces such as those involving IMEI or UUID identifiers. DEV URNs are expected to be a part of the IETF-provided basic URN types, covering identifiers that have previously not been possible to use in URNs.

3.2. Syntax

The identifier is expressed in ASCII characters and has a hierarchical structure as follows:

```
devurn = "urn:dev:" body componentpart
body = macbody / owbody / orgbody / osbody / opsbody / otherbody
macbody = %s"mac:" hexstring
owbody = %s"ow:" hexstring
orgbody = %s"org:" posnumber "-" identifier *( ":" identifier )
osbody = %s"os:" posnumber "-" serial *( ":" identifier )
opsbody = %s"ops:" posnumber "-" product "-" serial
          *( ":" identifier )
otherbody = subtype ":" identifier *( ":" identifier )
subtype = LALPHA *(DIGIT / LALPHA)
identifier = 1*devunreserved
identifiernodash = 1*devunreservednodash
product = identifiernodash
serial = identifier
componentpart = *( "_" identifier )
devunreservednodash = ALPHA / DIGIT / "."
devunreserved = devunreservednodash / "-"
hexstring = 1*(hexdigit hexdigit)
hexdigit = DIGIT / "a" / "b" / "c" / "d" / "e" / "f"
posnumber = NZDIGIT *DIGIT
ALPHA =  %x41-5A / %x61-7A
LALPHA =  %x41-5A
NZDIGIT = %x31-39
DIGIT =  %x30-39
```

The above syntax is represented in Augmented Backus-Naur Form (ABNF) as defined in [RFC5234] and [RFC7405]. The syntax also copies the DIGIT and ALPHA rules originally defined in [RFC5234], exactly as defined there.

The device identifier namespace includes five subtypes (see Section 4), and more may be defined in the future as specified in Section 7.

The optional underscore-separated components at the end of the DEV URN depict individual aspects of a device. The specific strings and their semantics are up to the designers of the device but could be used to refer to specific interfaces or functions within the device.

With the exception of the MAC address and 1-Wire DEV URNs, each DEV URN may also contain optional colon-separated identifiers. These are provided for extensibility.

There are no special character encoding rules or considerations for conforming with the URN syntax beyond those applicable for URNs in general [RFC8141] or the context where these URNs are carried (e.g., inside JSON [RFC8259] or SenML [RFC8428]). Due to the SenML rules in [RFC8428], Section 4.5.1, it is not desirable to use percent-encoding in DEV URNs, and the subtypes defined in this specification do not really benefit from percent-encoding. However, this specification does not deviate from the general syntax of URNs or their processing and normalization rules as specified in [RFC3986] and [RFC8141].

DEV URNs do not use r-, q-, or f-components as defined in [RFC8141].

Specific subtypes of DEV URNs may be validated through mechanisms discussed in Section 4.

The string representation of the device identifier URN is fully compatible with the URN syntax.

3.2.1. Character Case and URN-Equivalence

The DEV URN syntax allows both uppercase and lowercase characters. The URN-equivalence of the DEV URNs is defined per [RFC8141], Section 3.1, i.e., two URNs are URN-equivalent if their assigned-name portions are octet-by-octet equal after applying case normalization to the URI scheme ("urn") and namespace identifier ("dev"). The rest of the DEV URN is compared in a case-sensitive manner. It should be noted that URN-equivalence matching merely quickly shows that two URNs are definitely the same for the purposes of caching and other similar uses. Two DEV URNs may still refer to the same entity and may not be found to be URN-equivalent according to the [RFC8141] definition. For instance, in ABNF, strings are case insensitive (see [RFC5234], Section 2.3), and a MAC address could be represented either with uppercase or lowercase hexadecimal digits.

Character case is not otherwise significant for the DEV URN subtypes defined in this document. However, future subtypes might include identifiers that use encodings such as base64, which encodes strings in a larger variety of characters and might even encode binary data.

To facilitate equivalence checks, it is RECOMMENDED that implementations always use lowercase letters where they have a choice in case, unless there is a reason otherwise. (Such a reason might be, for instance, the use of a subtype that requires the use of both uppercase and lowercase letters.)

3.3. Assignment

The process for identifier assignment is dependent on the used subtype and is documented in the specific subsection under Section 4.

Device identifiers are generally expected to identify a unique device, barring the accidental issue of multiple devices with the same identifiers. In many cases, device identifiers can also be changed by users or are sometimes assigned in an algorithmic or local fashion. Any potential conflicts arising from such assignments are not something that the DEV URNs as such manage; they simply are there to refer to a particular identifier. And, of course, a single device may (and often does) have multiple identifiers, e.g., identifiers associated with different link technologies it supports.

The DEV URN type SHOULD only be used for hardware-based identifiers that are expected to be persistent (with some limits, as discussed above).

3.4. Security and Privacy

As discussed in Section 6, care must be taken in the use of device-identifier-based identifiers due to their nature as long-term identifiers that are not normally changeable. Leakage of these identifiers outside systems where their use is justified should be controlled.

3.5. Interoperability

There are no specific interoperability concerns.

3.6. Resolution

The device identifiers are not expected to be globally resolvable. No identifier resolution system is expected. Systems may perform local matching of identifiers to previously seen identifiers or configured information, however.

3.7. Documentation

See RFC 9039.

3.8. Additional Information

See Section 1 for a discussion of related namespaces.

3.9. Revision Information

This is the first version of this registration.

4. DEV URN Subtypes

4.1. MAC Addresses

DEV URNs of the "mac" subtype are based on the EUI-64 identifier [IEEE.EUI64] derived from a device with a built-in 64-bit EUI-64. The EUI-64 is formed from 24 or 36 bits of organization identifier followed by 40 or 28 bits of device-specific extension identifier assigned by that organization.

In the DEV URN "mac" subtype, the hexstring is simply the full EUI-64 identifier represented as a hexadecimal string. It is always exactly 16 characters long.

MAC-48 and EUI-48 identifiers are also supported by the same DEV URN subtype. To convert a MAC-48 address to an EUI-64 identifier, the Organizationally Unique Identifier (OUI) of the MAC-48 address (the first three octets) becomes the organization identifier of the EUI-64 (the first three octets). The fourth and fifth octets of the EUI are set to the fixed value 0xffff (hexadecimal). The last three octets of the MAC-48 address become the last three octets of the EUI-64. The same process is used to convert an EUI-48 identifier, but the fixed value 0xfffe is used instead.

Identifier assignment for all of these identifiers rests within the IEEE Registration Authority.

Note that where randomized MAC addresses are used, the resulting DEV URNs cannot be expected to have uniqueness, as discussed in Section 3.3.

4.2. 1-Wire Device Identifiers

The 1-Wire system is a device communications bus system designed by Dallas Semiconductor Corporation. (1-Wire is a registered trademark.) 1-Wire devices are identified by a 64-bit identifier that consists of an 8-bit family code, a 48-bit identifier unique within a family, and an 8-bit Cyclic Redundancy Check (CRC) code [OW].

In DEV URNs with the "ow" subtype, the hexstring is a representation of the full 64-bit identifier as a hexadecimal string. It is always exactly 16 characters long. Note that the last two characters represent the 8-bit CRC code. Implementations MAY check the validity of this code.

Family code and identifier assignment for all 1-Wire devices rests with the manufacturers.

4.3. Organization-Defined Identifiers

Device identifiers that have only a meaning within an organization can also be used to represent vendor-specific or experimental identifiers or identifiers designed for use within the context of an organization.

Organizations are identified by their Private Enterprise Number (PEN) [RFC2578]. These numbers can be obtained from IANA. Current PEN assignments can be viewed at <https://www.iana.org/assignments/

enterprise-numbers/>, and new assignments are requested at <https://pen.iana.org/pen/PenApplication.page>.

Note that when included in an "org" DEV URN, the number cannot be zero or have leading zeroes, as the ABNF requires the number to start with a non-zero digit.

4.4. Organization Serial Numbers

The "os" subtype specifies an organization and serial number. Organizations are identified by their PEN. As with the organization-defined identifiers (Section 4.3), PEN number assignments are maintained by IANA, and assignments for new organizations can be made easily.

> Historical note: The "os" subtype was originally defined in the Open Mobile Alliance "Lightweight Machine to Machine" standard [LwM2M] but has been incorporated here to collect all syntaxes associated with DEV URNs in one place. At the same time, the syntax of this subtype was changed to avoid the possibility of characters that are not allowed in the SenML Name field (see [RFC8428], Section 4.5.1).

Organization serial number DEV URNs consist of the PEN number and the serial number. As with other DEV URNs, for carrying additional information and extensibility, optional colon-separated identifiers and underscore-separated components may also be included. The serial numbers themselves are defined by the organization, and this specification does not specify how they are allocated.

Organizations are also encouraged to select serial number formats that avoid the possibility of ambiguity in the form of leading zeroes or otherwise.

4.5. Organization Product and Serial Numbers

The DEV URN "ops" subtype was originally defined in the LwM2M standard but has been incorporated here to collect all syntaxes associated with DEV URNs in one place. The "ops" subtype specifies an organization, product class, and a serial number. Organizations are identified by their PEN. Again, as with the organization-defined identifiers (Section 4.3), PEN number assignments are maintained by IANA.

> Historical note: As with the "os" subtype, the "ops" subtype was originally defined in the Open Mobile Alliance "Lightweight Machine to Machine" standard [LwM2M].

Organization product and serial number DEV URNs consist of the PEN number, product class, and the serial number. As with other DEV URNs, for carrying additional information and extensibility, optional colon-separated identifiers and underscore-separated components may also be included. Both the product class and serial numbers themselves are defined by the organization, and this specification does not specify how they are allocated.

Organizations are also encouraged to select product and serial number formats that avoid possibility for ambiguity.

4.6. Future Subtypes

Additional subtypes may be defined in future specifications. See Section 7.

The DEV URN "example" subtype is reserved for use in examples. It has no specific requirements beyond those expressed by the ABNF in Section 3.2.

5. Examples

The following provides some examples of DEV URNs:

URN	Description
urn:dev:mac:0024beffff804ff1	The MAC-48 address of 0024be804ff1, converted to EUI-64 format
urn:dev:mac:0024befffe804ff1	The EUI-48 address of 0024be804ff1, converted to EUI-64 format
urn:dev:mac:acde48234567019f	The EUI-64 address of acde48234567019f
urn:dev:ow:10e2073a01080063	A 1-Wire temperature sensor
urn:dev:ow:264437f5000000ed_humidity	The humidity part of a multi-sensor device
urn:dev:ow:264437f5000000ed_temperature	The temperature part of a multi-sensor device
urn:dev:org:32473-foo	An organization-specific URN in the example organization 32473 in [RFC5612]
urn:dev:os:32473-123456	Device 123456 in the example organization in [RFC5612]
urn:dev:os:32473-12-34-56	A serial number with dashes in it
urn:dev:ops:32473-Refrigerator-5002	Refrigerator serial number 5002 in the

	example organization in [RFC5612]
urn:dev:example:new-1-2-3_comp	An example of something that is not defined today, and is not one of the mac, ow, os, or ops subtypes

Table 1

The DEV URNs themselves can then appear in various contexts. A simple example of this is the use of DEV URNs in SenML data. This example from [RFC8428] shows a measurement from a 1-Wire temperature gauge encoded in the JSON syntax:

```
[
  {"n":"urn:dev:ow:10e2073a01080063","u":"Cel","v":23.1}
]
```

6. Security Considerations

On most devices, the user can display device identifiers. Depending on circumstances, device identifiers may or may not be modified or tampered with by the user. An implementation of the DEV URN MUST preserve such limitations and behaviors associated with the device identifiers. In particular, a device identifier that is intended to be immutable should not become mutable as a part of implementing the DEV URN type. More generally, nothing in this document should be construed to override what the relevant device specifications have already said about the identifiers.

6.1. Privacy

Other devices in the same network may or may not be able to identify the device. For instance, on an Ethernet network, the MAC address of a device is visible to all other devices.

DEV URNs often represent long-term stable unique identifiers for devices. Such identifiers may have privacy and security implications because they may enable correlating information about a specific device over a long period of time, location tracking, and device-specific vulnerability exploitation [RFC7721]. Also, in some systems, there is no easy way to change the identifier. Therefore, these identifiers need to be used with care, and special care should be taken to avoid leaking identifiers outside of the system that is intended to use them.

6.2. Validity

Information about identifiers may have significant effects in some applications. For instance, in many sensor systems, the identifier information is used for deciding how to use the data carried in a measurement report. In some other systems, identifiers may be used in policy decisions.

It is important that systems be designed to take into account the possibility of devices reporting incorrect identifiers (either accidentally or maliciously) and the manipulation of identifiers in communications by illegitimate entities. Integrity protection of communications or data objects, the use of trusted devices, and various management practices can help address these issues.

Similar to the advice in [RFC4122], Section 6: Do not assume that DEV URNs are hard to guess.

7. IANA Considerations

Per this document, IANA has registered a new URN namespace for "dev", as described in Section 3.

IANA has created a "DEV URN Subtypes" registry under "Device Identification". The initial values in this registry are as follows:

Subtype	Description	Reference
mac	MAC Addresses	RFC 9039, Section 4.1
ow	1-Wire Device Identifiers	RFC 9039, Section 4.2
org	Organization-Defined Identifiers	RFC 9039, Section 4.3
os	Organization Serial Numbers	RFC 9039, Section 4.4
ops	Organization Product and Serial Numbers	RFC 9039, Section 4.5
example	Reserved for examples	RFC 9039, Section 4.6

Table 2

Additional subtypes for DEV URNs can be defined through Specification Required or IESG Approval [RFC8126]. These allocations are appropriate when there is a new namespace of some type of device identifier that is defined in a stable fashion and has a publicly available specification.

Note that the organization (Section 4.3) device identifiers can also be used in some cases, at least as a temporary measure. It is preferable, however, that long-term usage of a broadly employed device identifier be registered with IETF rather than used through the organization device identifier type.

8. References

8.1. Normative References

[IEEE.EUI64] IEEE, "Guidelines for Use of Extended Unique Identifier (EUI), Organizationally Unique Identifier (OUI), and Company ID (CID)", August 2017, <https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/tutorials/eui.pdf>.

[OW] Maxim, "Guide to 1-Wire Communication", June 2008, <https://www.maximintegrated.com/en/design/technical-documents/tutorials/1/1796.html>.

[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>.

[RFC2578] McCloghrie, K., Ed., Perkins, D., Ed., and J. Schoenwaelder, Ed., "Structure of Management Information Version 2 (SMIv2)", STD 58, RFC 2578, DOI 10.17487/RFC2578, April 1999, <https://www.rfc-editor.org/info/rfc2578>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017, <https://www.rfc-editor.org/info/rfc8141>.

[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>.

8.2. Informative References

[CoRE-RD] Amsüss, C., Ed., Shelby, Z., Koster, M., Bormann, C., and P. van der Stok, "CoRE Resource Directory", Work in Progress, Internet-Draft, draft-ietf-core-resource-directory-28, 7 March 2021, <https://datatracker.ietf.org/doc/html/draft-ietf-core-resource-directory-28>.

[LwM2M] Alliance, O. M., "OMA Lightweight Machine to Machine

Requirements", OMA Standard Candidate Version 1.2, January 2019, <https://www.openmobilealliance.org/release/LightweightM2M/V1_2-20190124-C/OMA-RD-LightweightM2M-V1_2-20190124-C.pdf>.

[RFC3261] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and E. Schooler, "SIP: Session Initiation Protocol", RFC 3261, DOI 10.17487/RFC3261, June 2002, <https://www.rfc-editor.org/info/rfc3261>.

[RFC4122] Leach, P., Mealling, M., and R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace", RFC 4122, DOI 10.17487/RFC4122, July 2005, <https://www.rfc-editor.org/info/rfc4122>.

[RFC5612] Eronen, P. and D. Harrington, "Enterprise Number for Documentation Use", RFC 5612, DOI 10.17487/RFC5612, August 2009, <https://www.rfc-editor.org/info/rfc5612>.

[RFC6920] Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B., Keranen, A., and P. Hallam-Baker, "Naming Things with Hashes", RFC 6920, DOI 10.17487/RFC6920, April 2013, <https://www.rfc-editor.org/info/rfc6920>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7252] Shelby, Z., Hartke, K., and C. Bormann, "The Constrained Application Protocol (CoAP)", RFC 7252, DOI 10.17487/RFC7252, June 2014, <https://www.rfc-editor.org/info/rfc7252>.

[RFC7254] Montemurro, M., Ed., Allen, A., McDonald, D., and P. Gosden, "A Uniform Resource Name Namespace for the Global System for Mobile Communications Association (GSMA) and the International Mobile station Equipment Identity (IMEI)", RFC 7254, DOI 10.17487/RFC7254, May 2014, <https://www.rfc-editor.org/info/rfc7254>.

[RFC7405] Kyzivat, P., "Case-Sensitive String Support in ABNF", RFC 7405, DOI 10.17487/RFC7405, December 2014, <https://www.rfc-editor.org/info/rfc7405>.

[RFC7540] Belshe, M., Peon, R., and M. Thomson, Ed., "Hypertext Transfer Protocol Version 2 (HTTP/2)", RFC 7540, DOI 10.17487/RFC7540, May 2015, <https://www.rfc-editor.org/info/rfc7540>.

[RFC7721] Cooper, A., Gont, F., and D. Thaler, "Security and Privacy Considerations for IPv6 Address Generation Mechanisms", RFC 7721, DOI 10.17487/RFC7721, March 2016, <https://www.rfc-editor.org/info/rfc7721>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

[RFC8428] Jennings, C., Shelby, Z., Arkko, J., Keranen, A., and C. Bormann, "Sensor Measurement Lists (SenML)", RFC 8428, DOI 10.17487/RFC8428, August 2018, <https://www.rfc-editor.org/info/rfc8428>.

[RFC8464] Atarius, R., "A URN Namespace for Device Identity and Mobile Equipment Identity (MEID)", RFC 8464, DOI 10.17487/RFC8464, September 2018, <https://www.rfc-editor.org/info/rfc8464>.

[W3C.REC-xml-19980210] Sperberg-McQueen, C., Bray, T., and J. Paoli, "Extensible Markup Language (XML) 1.0", W3C Recommendation, February 1998, <http://www.w3.org/TR/1998/REC-xml-19980210>.

Acknowledgments

The authors would like to thank Ari Keränen, Stephen Farrell, Christer Holmberg, Peter Saint-Andre, Wouter Cloetens, Jaime Jimenez, Joseph Knapp, Padmakumar Subramani, Mert Ocak, Hannes Tschofenig, Jim Schaad, Thomas Fossati, Carsten Bormann, Marco Tiloca, Barry Leiba, Amanda Baber, Juha Hakala, Dale Worley, Warren Kumari, Benjamin Kaduk, Brian Weis, John Klensin, Dave Thaler, Russ Housley, Dan Romascanu, Éric Vyncke, Roman Danyliw, and Ahmad Muhanna for their feedback and interesting discussions in this problem space. We would also like to note prior documents that focused on specific device identifiers, such as [RFC7254] and [RFC8464].

Authors' Addresses

Jari Arkko
Ericsson
FI-02420 Jorvas
Finland

Email: jari.arkko@piuha.net

Cullen Jennings
Cisco
170 West Tasman Drive
San Jose, CA 95134
United States of America

Phone: +1 408 421-9990
Email: fluffy@iii.ca

Zach Shelby
Edge Impulse
3031 Tisch Way

San Jose, CA 95128
United States of America

Email: zach@edgeimpulse.com