Workgroup: Network Working Group
Internet-Draft:
draft-irtf-t2trg-secure-bootstrapping-02
Published: 25 April 2022
Intended Status: Informational
Expires: 27 October 2022
Authors: M. Sethi
Aalto University
B. Sarikaya
Denpel Informatique
D. Garcia-Carrillo
University of Oviedo

Terminology and processes for initial security setup of IoT devices

Abstract

This document provides an overview of terms that are commonly used when discussing the initial security setup of Internet of Things (IoT) devices. This document also presents a brief but illustrative survey of protocols and standards available for initial security setup of IoT devices. For each protocol, we identify the terminology used, the entities involved, the initial assumptions, the processes necessary for completetion, and the knowledge imparted to the IoT devices after the setup is complete.

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 27 October 2022.

Table of Contents

1. Introduction

Initial security setup for a device refers to any process that takes place before a device can become operational. The phrase "initial security setup" intentionally includes the term "security" as setup of devices without adequate security or with insecure processes is no longer acceptable. The initial security setup process, among other things, involves network discovery and selection, access authentication, configuration of necessary credentials and parameters.

Initial security setup for IoT devices is challenging because the size of an IoT network varies from a couple of devices to tens of thousands, depending on the application. Moreover, devices in IoT networks are produced by a variety of vendors and are typically heterogeneous in terms of the constraints on their power supply,

communication capability, computation capacity, and user interfaces available. This challenge of initial security setup in IoT was identified by Sethi et al. [Sethi14] while developing a solution for smart displays.

Initial security setup of devices typically also involves providing them with some sort of network connectivity. The functionality of a disconnected device is rather limited. Initial security setup of devices often assumes that some network has been setup a-priori. Setting up and maintaining a network itself is challenging. For example, users may need to configure the network name (called as Service Set Identifier (SSID) in Wi-Fi networks) and passpharse before new devices can be setup. Specifications such as the Wi-Fi Alliance Simple Configuration [simpleconn] help users setup networks. However, this document is only focused on terminology and processes associated with initial security setup of devices and excludes any discussion on setting up networks.

There are several terms that are used in the context of initial security setup of devices:

*Bootstrapping

*Provisioning

*Onboarding

*Enrollment

*Commissioning

*Initialization

*Configuration

*Registration

*Discovery

In addition to using a variety of different terms, initial security setup mechanisms can rely on a number of entities. For example, a companion smartphone device maybe necessary for some initial security setup mechanisms. Moreover, security setup procedures have diverese initial assumptions about the device being setup. As an example, an initial security setup mechanism may assume that the device is provisioned with a pre-shared key and a list of trusted network identifiers. Finally, initial security setup mechanisms impart different information to the device after completion. For example, some mechanisms may only provide a key for use with an

authorization server, while others may configure elaborate access control lists directly.

The next section provides an overview of some standards and protocols for initial security setup of IoT devices. For each mechanism, the following are explicitly identified:

- *Terminology used
- *Entities or "players" involved
- *Initial assumptions about the device
- *Processes necessary for completetion
- *Knowledge imparted to the device after completion

2. Standards and Protocols

2.1. Device Provisioning Protocol (DPP)

The Wi-Fi Alliance Device provisioning protocol (DPP) [dpp] describes itself as a standardized protocol for providing user friendly Wi-Fi setup while maintaining or increasing the security. DPP relies on a configurator, e.g. a smartphone application, for setting up all other devices, called enrollees, in the network. DPP has the following three phases/sub-protocols:

- *Bootstrapping: The configurator obtains bootstrapping information from the enrollee using an out-of-band channel such as scanning a QR code or tapping NFC. The bootstrapping information includes the public-key of the device and metadata such as the radio channel on which the device is listening.
- *Authentication: In DPP, either the configurator or the enrollee can initiate the authentication protocol. The side initiating the authentication protocol is called as the initiator while the other side is called the responder. The authentication sub-protocol provides authentication of the responder to an initiator. It can optionally authenticate the initiator to the responder (only if the bootstrapping information was exchange out-of-band in both directions).
- *Configuration: Using the key established from the authentication protocol, the enrollee asks the configurator for network information such as the SSID and passphrase of the access point.

DPP has the following characteristics:

*Terms: Bootstrapping, configuration, discovery, enrollment, provisioning.

*Players: Authenticator, Bootstrap Server, Client, Configurator, Device, Initiator, Manager, Manufacturer, Owner, Peer, Peer, Persona, Responder, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.2. Open Mobile Alliance (OMA) Lightweight M2M (LwM2M)

The OMA LwM2M specification [oma] defines an architecture where a new device (LwM2M client) contacts a Bootstrap-server which is responsible for provisioning essential information such as credentials. After receiving this essential information, the LwM2M client device registers itself with one or more LwM2M Servers which will manage the device during its lifecycle. The current standard defines the following four bootstrapping modes:

*Factory Bootstrap: An IoT device in this case is configured with all the necessary bootstrap information during manufacturing and prior to its deployment.

*Bootstrap from Smartcard: An IoT device retrieves and processes all the necessary bootstrap data from a Smartcard.

*Client Initiated Bootstrap: This mode provides a mechanism for an IoT client device to retrieve the bootstrap information from a Bootstrap Server. This requires the client device to have an account at the Bootstrap Server and credentials to obtain the necessary information securely.

*Server Initiated Bootstrap: In this bootstrapping mode, the bootstrapping server configures all the bootstrap information on the IoT device without receiving a request from the client. This means that the bootstrap server needs to know if a client IoT Device is ready for bootstrapping before it can be configured. For example, a network may inform the bootstrap server of a new connecting IoT client device.

OMA has the following characteristics:

*Terms: Bootstrapping, provisioning, intialization, configuration, registration.

*Players: Bootstrap Server, Client, Device, Manufacturer, Owner, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.3. Open Connectivity Foundation (OCF)

The Open Connectivity Foundation (OCF) [ocf] defines the process before a device is operational as onboarding. The first step of this onboarding process is configuring the ownership, i.e., establishing a legitimate user that owns the device. For this, the user is supposed to use an Onboarding tool (OBT) and an Owner Transfer Method (OTM).

The OBT is described as a logical entity that may be implemented on a single or multiple entities such as a home gateway, a device management tool, etc. OCF lists several optional OTMs. At the end of the execution of an OTM, the onboarding tool must have provisioned an Owner Credential onto the device. The following owner transfer methods are specified:

*Just works: Performs an un-authenticated Diffie-Hellman key exchange over Datagram Transport Layer Security (DTLS). The key exchange results in a symmetric session key which is later used for provisioning. Naturally, this mode is vulnerable to on-path attackers.

*Random PIN: The device generates a PIN code that is entered into the onboarding tool by the user. This pin code is used together with TLS-PSK ciphersuites for establishing a symmetric session key. OCF recommends PIN codes to have an entropy of 40 bits.

*Manufacturer certificate: An onboarding tool authenticates the device by verifying a manufacturer installed certificate. Similarly, the device may authenticate the onboarding tool by verifying its signature.

*Vendor specific: Vendors implement their own transfer method that accommodates any specific device constraints.

Once the onboarding tool and the new device have authenticated and established secure communication, the onboarding tool provisions/ configures the device with Owner credentials. Owner credentials may consist of certificates, shared keys, or Kerberos tickets for example.

The OBT additionally configures/provisions information about the Access Management Service (AMS), the Credential Management Service (CMS), and the credentials for interacting with them. The AMS is responsible for provisioning access control entries, while the CMS provisions security credentials necessary for device operation.

OCF has the following characteristics:

*Terms: Configuration, discovery, enrollment, onboarding, provisioning, registration,

*Players: Client, Device, Manager, Manufacturer, Owner, Peer, Responder, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.4. Bluetooth

Bluetooth mesh specifies a provisioning protocol. The goal of the provisioning phase is to securely incorporate a new Bluetooth mesh node, by completing two critical tasks. First, to authenticate the unprovisioned device and second, to create a secure link with said device to share information.

The provisioning process is divided into five distinct stages summarize next:

*Beaconing for discover: The new unprovisioned device is discovered by the provisioner

*Negotiation: The unprovisioned device indicates to the provisioner a set of capabilities such as the security algorithms supported, the availability of its public key using an Out-of-Band (OOB) channel and the input/output interfaces supported

*Public-key exchange: The authentication method is selected by the provisioner and both devices exchange Elliptic-curve Diffie-Hellman (ECDH) public keys. These keys may be static or ephemeral. Their exchange can be done in two ways, either via Out-of-Band or directly through a Bluetooth link. Each device then generates a symmetric key, named ECDHSecret, by combining its own private key and the public key of the peer device. The EDCHSecret is used to protect communication between the two devices.

*Authentication: During this phase, the ECDH key exchange is authenticated. The authentication method can be Output OOB, Input OOB, static OOB, or No OOB. With Output OOB, the unprovisioned device chooses a random number and outputs that number in manner consistent with its capabilities. The provisioner then inputs this number. Then, a check confirmation value operation is performed. This involves a cryptographic exchange regarding (in this case) the random number to complete the authentication. With Input OOB, the roles are reversed, being the provisioner the entity that generates the random number. When neither of the previous authentication procedures are feasible, both the provisioner and unprovisioned device generate a random number and require the user supervising the setup to verify that values on the device and provisioner are the same.

*Distribution of provisioning data: At this point, the provisioning process can be protected. This involves the distribution of data such as a Network key, to secure the communications at network layer and a unicast address among other information.

Bluetooth mesh has the following characteristics:

*Terms: Configuration, discovery, provisioning.

*Players: Client, Device, Manager, Manufacturer, Peer, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.5. Fast IDentity Online (FIDO) alliance

The Fast IDentity Online Alliance (FIDO) is currently specifying an automatic onboarding protocol for IoT devices [fidospec]. The goal of this protocol is to provide a new IoT device with information for interacting securely with an online IoT platform. This protocol allows owners to choose the IoT platform for their devices at a late stage in the device lifecyle. The draft specification refers to this feature as "late binding".

The FIDO IoT protocol itself is composed of one Device Initialization (DI) protocol and 3 Transfer of Ownership (TO) protocols TO0, TO1, TO2. Protocol messages are encoded in Concise Binary Object Representation (CBOR) [RFC8949] and can be transported over application layer protocols such as Constrained Application Protocol (CoAP) [RFC7252] or directly over TCP, Bluetooth etc. FIDO

IoT however assumes that the device already has IP connectivity to a rendezvous server. Establishing this initial IP connectivity is explicitly stated as "out-of-scope" but the draft specification hints at the usage of Hypertext Transfer Protocol (HTTP) [RFC7230] proxies for IP networks and other forms of tunneling for non-IP networks.

The specification only provides a non-normative example of the DI protocol which must be executed in the factory during device manufacture. This protocol embeds initial ownership and manufacturing credentials into Restricted Operation Environment (ROE) on the device. The initial information embedded also includes a unique device identifier (called as GUID in the specification). After DI is executed, the manufacturer has an ownership voucher which is passed along the supply chain to the device owner.

When a device is unboxed and powered on by the new owner, the device discovers a network-local or an Internet-based rendezvous server. Protocols (TO0, TO1, and TO2) between the device, the rendezvous server, and the new owner (as the owner onboarding service) ensure that the device and the new owner are able to authenticate each other. Thereafter, the new owner establishes cryptographic control of the device and provides it with credentials of the IoT platform which the device should used.

FIDO has the following characteristics:

*Terms: Provisioning, onboarding, commissioning, initialization.

*Players: Device, Manager, Manufacturer, Owner, Rendezvous Server, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.6. Enrollment over Secure Transport (EST)

Enrollment over Secure Transport (EST) [RFC7030] defines a profile of Certificate Management over CMS (CMC) [RFC5272]. EST relies on Hypertext Transfer Protocol (HTTP) and Transport Layer Security (TLS) for exchanging CMC messages and allows client devices to obtain client certificates and associated Certification Authority (CA) certificates. A companion specification for using EST over secure CoAP has also been standardized [I-D.ietf-ace-coap-est]. EST assumes that some initial information is already distributed so that EST client and servers can perform mutual authentication before continuing with protocol. [RFC7030] further defines "Bootstrap

Distribution of CA Certificates" which allows minimally configured EST clients to obtain initial trust anchors. It relies on human users to verify information such as the CA certificate "fingerprint" received over the unauthenticated TLS connection setup. After successful completion of this bootstrapping step, clients can proceed to the enrollment step during which they obtain client certificates and associated CA certificates.

EST has the following characteristics:

*Terms: Bootstrapping, enrollment, initialization, configuration.

*Players: Administrator, Client, Device, Manufacturer, Owner, Peer, Peer, Responder, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.7. Bootstrapping Remote Secure Key Infrastructures (BRSKI)

The ANIMA working group is working on a bootstrapping solution for devices that relies on 802.1AR vendor certificates called Bootstrapping Remote Secure Key Infrastructures (BRSKI) [RFC8995]. In addition to vendor installed IEEE 802.1AR certificates, a vendor based service on the Internet is required. Before being authenticated, a new device only needs link-local connectivity, and does not require a routable address. When a vendor provides an Internet based service, devices can be forced to join only specific domains. The document highlights that the described solution is aimed in general at non-constrained (i.e. class 2+ defined in [RFC7228]) devices operating in a non-challenged network. It claims to scale to thousands of devices located in hostile environments, such as ISP provided CPE devices which are drop-shipped to the end user.

BRSKI has the following characteristics:

*Terms: Bootstrapping, provisioning, enrollment, onboarding.

*Players: Administrator, Client, Cloud Registrar, Configurator, Device, Domain Registrar, Initiator, Join Proxy, JRC, Manufacturer, Owner, Peer, Pledge, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.8. Secure Zero Touch Provisioning

[RFC8572] defines a bootstrapping strategy for enabling devices to securely obtain all the configuration information with no installer input, beyond the actual physical placement and connection of cables. Their goal is to enable a secure NETCONF [RFC6241] or RESTCONF [RFC8040] connection to the deployment specific network management system (NMS). This bootstrapping method requires the devices to be configured with trust anchors in the form of X.509 certificates. [RFC8572] is similar to BRSKI based on [RFC8366].

SZTP has the following characteristics:

*Terms: Bootstrapping, provisioning, onboarding, enrollment, configuration, discovery.

*Players: Administrator, Bootstrap Server, Client, Device, Manufacturer, Onboarding Server, Owner, Redirect Server, Responder, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.9. Nimble out-of-band authentication for EAP (EAP-NOOB)

EAP-NOOB [RFC9140] defines an EAP method where the authentication is based on a user-assisted out-of-band (OOB) channel between the server and peer. It is intended as a generic bootstrapping solution for IoT devices which have no pre-configured authentication credentials and which are not yet registered on the authentication server. This method claims to be more generic than most ad-hoc bootstrapping solutions in that it supports many types of OOB channels. The exact in-band messages and OOB message contents are specified and not the OOB channel details. EAP-NOOB also supports IoT devices with only output (e.g. display) or only input (e.g. camera). It makes combined use of both secrecy and integrity of the OOB channel for more robust security than the ad-hoc solutions.

EAP-NOOB has the following characteristics:

*Terms: Bootstrapping, configuration, registration.

*Players: Administrator, Authenticator, Client, Device, Manufacturer, Owner, Peer, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.10. LPWAN

Low Power Wide Area Network (LPWAN) encompasses a wide variety of technologies whose link-layer characteristics are severely constrained in comparison to other typical IoT link-layer technologies such as Bluetooth or IEEE 802.15.4. While some LPWAN technologies rely on proprietary bootstrapping solutions which are not publicly accessible, others simply ignore the challenge of bootstrapping and key distribution. In this section, we discuss the bootstrapping methods used by LPWAN technologies covered in [RFC8376].

*LoRaWAN [LoRaWAN] describes its own protocol to authenticate nodes before allowing them join a LoRaWAN network. This process is called as joining and it is based on pre-shared keys (called AppKeys in the standard). The joining procedure comprises only one exchange (join-request and join-accept) between the joining node and the network server. There are several adaptations to this joining procedure that allow network servers to delegate authentication and authorization to a backend AAA infrastructure [RFC2904].

*Wi-SUN Alliance Field Area Network (FAN) uses IEEE 802.1X and EAP-TLS for network access authentication. It performs a 4-way handshake to establish a session keys after EAP-TLS authentication.

*NB-IoT relies on the traditional 3GPP mutual authentication scheme based on a shared-secret in the Subscriber Identity Module (SIM) of the device and the mobile operator.

*Sigfox security is based on unique device identifiers and cryptographic keys. As stated in [RFC8376], although the algorithms and keying details are not publicly available, there is sufficient information to indicate that bootstrapping in Sigfox is based on pre-established credentials between the device and the Sigfox network.

From the above, it is clear that all LPWAN technologies rely on pre-provisioned credentials for authentication between a new device and the network.

LPWAN has the following characteristics:

*Terms: Bootstrapping, provisioning, configuration, discovery.

*Players: Administrator, Authenticator, Border Router, Client, Device, Manager, Network Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

2.11. Thread

Thread Group commissioning [threadcommissioning] introduces a two phased process i.e. Petitioning and Joining. Entities involved are leader, joiner, commissioner, joiner router, and border router. Leader is the first device in Thread network that must be commissioned using out-of-band process and is used to inject correct user generated Commissioning Credentials (can be changed later) into Thread Network. Joiner is the node that intends to get authenticated and authorized on Thread Network. Commissioner is either within the Thread Network (Native) or connected with Thread Network via a WLAN (External).

Under some topologies, Joiner Router and Border Router facilitate the Joiner node to reach Native and External Commissioner, respectively. Petitioning begins before Joining process and is used to grant sole commissioning authority to a Commissioner. After an authorized Commissioner is designated, eligible thread devices can join network. Pair-wise key is shared between Commissioner and Joiner, network parameters (such as network name, security policy, etc.,) are sent out securely (using pair-wise key) by Joiner Router to Joiner for letting Joiner to join the Thread Network. Entities involved in Joining process depends on system topology i.e. location of Commissioner and Joiner. Thread networks only operate using IPv6. Thread devices can devise GUAs (Global Unicast Addresses) [RFC4291]. Provision also exist via Border Router, for Thread device to acquire individual global address by means of DHCPv6 or using SLAAC (Stateless Address Autoconfiguration) address derived with advertised network prefix.

Thread has the following characteristics:

*Terms: Commissioning, discovery, provisioning.

*Players: Administrator, Border Agent, Border Router, Commissioner, Commissioner Candidate, Configurator, Device, End

Device, End Device, Endpoint Identifier, Initiator, Joiner, Joiner Router, Owner, Peer, Peer, Responder, Server, User

*Initial beliefs assumed in the device:

*Processes:

*Beliefs imparted to the device after protocol execution:

3. Comparison

There are several stages before a device becomes fully operational. This typically involves establishing some initial trust after which credentials and other parameters are configured. For DPP, bootstrapping is the first step before authentication and provisioning of credentials can occur. For EST, bootstrapping happens as the first step when the client devices have no certificates available for starting enrollment. Provisioning/ configuring are terms used for providing additional information to devices before they are fully operational. For example, credentials are provisioned onto the device. But before credential provisioning, a device is bootstrapped and authenticated. Some protocols may only deal with parts of the process. For example, TLS maybe used for authentication after bootstrapping. A separate device management protocol then may run over this TLS tunnel for provisioning operational information and credentials.

3.1. Comparison of terminology

3.2. Comparison of players

3.3. Comparison of initial beliefs

3.4. Comparison of processes

3.5. Comparison of knowledge imparted to the device

4. Security Considerations

This draft does not take any posture on the security properties of the different bootstrapping protocols discussed. Specific security considerations of bootstrapping protocols are present in the respective specifications.

Nonetheless, we briefly discuss some important security aspects which are not fully explored in various specifications.

Firstly, an IoT system may deal with authorization for resources and services separately from initial security setup in terms of timing as well as protocols. As an example, two resource-constrained

devices A and B may perform mutual authentication using credentials provided by an offline third-party X before device A obtains authorization for running a particular application on device B from an online third-party Y. In some cases, authentication and authorization maybe tightly coupled, e.g., successful authentication also means successful authorization.

Secondly, initial security setup of IoT devices may be necessary several times during the device lifecycle since keys have limited lifetimes and devices may be lost or resold. Protocols and systems must have adequate provisions for revocation and fresh security setup. A rerun of the security setup mechanism must be as secure as the initial security setup regardless of whether it is done manually or automatically over the network.

Lastly, some IoT networks use a common group key for multicast and broadcast traffic. As the number of devices in a network increase over time, a common group key may not be scalable and the same network may need to be split into separate groups with different keys. Bootstrapping and provisioning protocols may need appropriate mechanisms for identifying and distributing keys to the current member devices of each group.

5. IANA Considerations

There are no IANA considerations for this document.

6. Acknowledgements

We would like to thank Tuomas Aura, Hannes Tschofenig, and Michael Richardson for providing extensive feedback as well as Rafa Marin-Lopez for his support.

7. Informative References

[dpp] Wi-Fi Alliance, "Wi-Fi Device Provisioning Protocol (DPP)", Wi-Fi Alliance Specification version 1.1, 2018, <https://www.wi-fi.org/download.php?file=/sites/default/files/private/Device_Provisioning_Protocol_Specification_v1.1_1.pdf>.

[fidospec] Fast Identity Online Alliance, "FIDO Device Onboard Specification", Fido Alliance Version: 1.0, March 2021, <https://fidoalliance.org/specifications/download-iot-specifications/>.

[I-D.ietf-ace-coap-est] Stok, P. V. D., Kampanakis, P., Richardson, M. C., and S. Raza, "EST over secure CoAP (EST-coaps)", Work in Progress, Internet-Draft, draft-ietf-ace-coap-

est-18, 6 January 2020, <https://www.ietf.org/archive/id/draft-ietf-ace-coap-est-18.txt>.

[I-D.ietf-ace-wg-coap-eap] Marin-Lopez, R. and D. Garcia-Carrillo, "EAP-based Authentication Service for CoAP", Work in Progress, Internet-Draft, draft-ietf-ace-wg-coap-eap-06, 7 December 2021, <https://www.ietf.org/archive/id/draft-ietf-ace-wg-coap-eap-06.txt>.

[IEEE802.15.4] IEEE, "IEEE Standard for Low-Rate Wireless Networks", IEEE Std. 802.15.4-2015, April 2016, <http://standards.ieee.org/findstds/standard/802.15.4-2015.html>.

[LoRaWAN] LoRa Alliance, "LoRa Specification V1.1", LoRa Alliance Version: 1.1, October 2017, <https://lora-alliance.org/resource_hub/lorawan-specification-v1-1/>.

[ocf] Open Connectivity Foundation, "OCF Security Specification", Version 2.2.2, February 2021, <https://openconnectivity.org/specs/OCF_Security_Specification_v2.2.2.pdf>.

[oma] Open Mobile Alliance, "Lightweight Machine to Machine Technical Specification: Core", Approved Version 1.2, November 2020, <https://www.openmobilealliance.org/release/LightweightM2M/V1_2-20201110-A/OMA-TS-LightweightM2M_Core-V1_2-20201110-A.pdf>.

[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>.

[RFC2904] Vollbrecht, J., Calhoun, P., Farrell, S., Gommans, L., Gross, G., de Bruijn, B., de Laat, C., Holdrege, M., and D. Spence, "AAA Authorization Framework", RFC 2904, DOI 10.17487/RFC2904, August 2000, <https://www.rfc-editor.org/info/rfc2904>.

[RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, Ed., "Extensible Authentication Protocol (EAP)", RFC 3748, DOI 10.17487/RFC3748, June 2004, <https://www.rfc-editor.org/info/rfc3748>.

[RFC3971] Arkko, J., Ed., Kempf, J., Zill, B., and P. Nikander, "SEcure Neighbor Discovery (SEND)", RFC 3971, DOI 10.17487/RFC3971, March 2005, <https://www.rfc-editor.org/info/rfc3971>.

[RFC3972] Aura, T., "Cryptographically Generated Addresses (CGA)", RFC 3972, DOI 10.17487/RFC3972, March 2005, <https://www.rfc-editor.org/info/rfc3972>.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, DOI 10.17487/RFC4120, July 2005, <https://www.rfc-editor.org/info/rfc4120>.

[RFC4253] Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, DOI 10.17487/RFC4253, January 2006, <https://www.rfc-editor.org/info/rfc4253>.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, DOI 10.17487/RFC4291, February 2006, <https://www.rfc-editor.org/info/rfc4291>.

[RFC4764] Bersani, F. and H. Tschofenig, "The EAP-PSK Protocol: A Pre-Shared Key Extensible Authentication Protocol (EAP) Method", RFC 4764, DOI 10.17487/RFC4764, January 2007, <https://www.rfc-editor.org/info/rfc4764>.

[RFC5191] Forsberg, D., Ohba, Y., Ed., Patil, B., Tschofenig, H., and A. Yegin, "Protocol for Carrying Authentication for Network Access (PANA)", RFC 5191, DOI 10.17487/RFC5191, May 2008, <https://www.rfc-editor.org/info/rfc5191>.

[RFC5272] Schaad, J. and M. Myers, "Certificate Management over CMS (CMC)", RFC 5272, DOI 10.17487/RFC5272, June 2008, <https://www.rfc-editor.org/info/rfc5272>.

[RFC6241] Enns, R., Ed., Bjorklund, M., Ed., Schoenwaelder, J., Ed., and A. Bierman, Ed., "Network Configuration Protocol (NETCONF)", RFC 6241, DOI 10.17487/RFC6241, June 2011, <https://www.rfc-editor.org/info/rfc6241>.

[RFC7030] Pritikin, M., Ed., Yee, P., Ed., and D. Harkins, Ed., "Enrollment over Secure Transport", RFC 7030, DOI 10.17487/RFC7030, October 2013, <https://www.rfc-editor.org/info/rfc7030>.

[RFC7228] Bormann, C., Ersue, M., and A. Keranen, "Terminology for Constrained-Node Networks", RFC 7228, DOI 10.17487/RFC7228, May 2014, <https://www.rfc-editor.org/info/rfc7228>.

[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>.

[RFC7250] Wouters, P., Ed., Tschofenig, H., Ed., Gilmore, J., Weiler, S., and T. Kivinen, "Using Raw Public Keys in Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS)", RFC 7250, DOI 10.17487/RFC7250, June 2014, <https://www.rfc-editor.org/info/rfc7250>.

[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>.

[RFC7593] Wierenga, K., Winter, S., and T. Wolniewicz, "The eduroam Architecture for Network Roaming", RFC 7593, DOI 10.17487/RFC7593, September 2015, <https://www.rfc-editor.org/info/rfc7593>.

[RFC8040] Bierman, A., Bjorklund, M., and K. Watsen, "RESTCONF Protocol", RFC 8040, DOI 10.17487/RFC8040, January 2017, <https://www.rfc-editor.org/info/rfc8040>.

[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>.

[RFC8366] Watsen, K., Richardson, M., Pritikin, M., and T. Eckert, "A Voucher Artifact for Bootstrapping Protocols", RFC 8366, DOI 10.17487/RFC8366, May 2018, <https://www.rfc-editor.org/info/rfc8366>.

[RFC8376] Farrell, S., Ed., "Low-Power Wide Area Network (LPWAN) Overview", RFC 8376, DOI 10.17487/RFC8376, May 2018, <https://www.rfc-editor.org/info/rfc8376>.

[RFC8572] Watsen, K., Farrer, I., and M. Abrahamsson, "Secure Zero Touch Provisioning (SZTP)", RFC 8572, DOI 10.17487/RFC8572, April 2019, <https://www.rfc-editor.org/info/rfc8572>.

[RFC8949] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", STD 94, RFC 8949, DOI 10.17487/RFC8949, December 2020, <https://www.rfc-editor.org/info/rfc8949>.

[RFC8995] Pritikin, M., Richardson, M., Eckert, T., Behringer, M., and K. Watsen, "Bootstrapping Remote Secure Key Infrastructure (BRSKI)", RFC 8995, DOI 10.17487/RFC8995, May 2021, <https://www.rfc-editor.org/info/rfc8995>.

[RFC9140] Aura, T., Sethi, M., and A. Peltonen, "Nimble Out-of-Band Authentication for EAP (EAP-NOOB)", RFC 9140, DOI 10.17487/RFC9140, December 2021, <https://www.rfc-editor.org/info/rfc9140>.

[RFC9190] Preuß Mattsson, J. and M. Sethi, "EAP-TLS 1.3: Using the Extensible Authentication Protocol with TLS 1.3", RFC 9190, DOI 10.17487/RFC9190, February 2022, <https://www.rfc-editor.org/info/rfc9190>.

[Sethi14] Sethi, M., Oat, E., Di Francesco, M., and T. Aura, "Secure Bootstrapping of Cloud-Managed Ubiquitous Displays", Proceedings of ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp 2014), pp. 739-750, Seattle, USA, September 2014, <http://dx.doi.org/10.1145/2632048.2632049>.

[simpleconn] Wi-Fi Alliance, "Wi-Fi Simple Configuration", Version 2.0.7, 2019, <https://www.wi-fi.org/download.php?file=/sites/default/files/private/Wi-Fi_Simple_Configuration_Technical_Specification_v2.0.7.pdf>.

[threadcommissioning] Thread Group, "Thread Commissioning", 2015.

[vendorcert] IEEE std. 802.1ar-2009, "Standard for local and metropolitan area networks - secure device identity", December 2009.

Authors' Addresses

Mohit Sethi
Aalto University
FI-02150 Espoo
Finland

Email: mohit@iki.fi

Behcet Sarikaya
Denpel Informatique

Email: sarikaya@ieee.org

Dan Garcia-Carrillo
University of Oviedo
33207 Oviedo
Spain

Email: garciadan@uniovi.es