wirelessCONGRESS systems & applications

Authenticating Wireless Nodes in Building Automation: Challenges and Approaches

Aurelio Schellenbaum, Tobias Schläpfer, Christian Stauffer and Andreas Rüst
Zurich University of Applied Science (ZHAW)
Institute of Embedded Systems (InES)
Winterthur, Switzerland
andreas.ruest@zhaw.ch

Oskar Camenzind
Siemens Building Technologies
Zug, Switzerland
oskar.camenzind@siemens.com

***Abstract* — Modern wireless nodes in building automation systems interconnect natively through the Internet Protocol (IP). As a result, the emerging coalescence of existing IT networks with networks on the field level presents many challenges. Specifically, mutual authentication of devices in an IT environment is one of the main issues. Moreover, this mutual authentication has to take place with embedded devices in the field that feature manifold constraints and require a simple but secure provisioning. The Fairhair Alliance is in the process of standardizing an autonomic secure bootstrapping process to tackle these challenges. The paper outlines this automated approach and shows the successful implementation of a real-life prototype. This demonstrates that the required cryptographic functions and procedures are feasible on a constrained low power device.**

Keywords —IoT security, authentication, automated network enrollment, wireless building automation, low power wireless sensors, resource constrained systems

I. INTRODUCTION

Recent technologies and standards allow connecting constrained wireless nodes to the Internet by natively using the prevailing Internet Protocol (IP). Such standards include the protocol stack as defined by the Thread Group [1], based on CoAP, UDP, IPv6, 6LoWPAN and IEEE 802.15.4. As a result, the sensor and actuator networks on the field level will coalesced with the existing IT networks. Specifically, replacing gateways with routers significantly simplifies a building automation system and enables new applications. Employing IP communication, a central automation station can directly and uniformly access sensor and actuator services on field nodes.

Consequently, to become a full-fledged member of an IT domain, a constrained node on the field level has to fulfill specific security requirements. However, implementing such requirements is especially challenging on constrained low power and low cost nodes. Such nodes typically have decidedly lower resources with regard to compute performance, memory and network connectivity. Nevertheless, such nodes require a mutual authentication during the provisioning into an individual IT domain. Specifically, several trust relationships need to be established.

Before granting access to the node, the IT domain administrator requires proof that the node is not compromised, e.g. by loading malicious firmware. This proof includes not only the proof that the trusted supplier has manufactured the node but also a complete and unforgeable list of previous installations and owners. As building automation systems typically are an integral part of a building, they represent capital assets and change ownership during their lifetime. On the other hand, before legitimately joining a new domain, the individual node needs to know: Is the deployment into this specific building legit? The scale of building automation systems in large buildings with hundreds of nodes mandates a highly automated authentication process. A simple provisioning of the nodes is essential.

Funded by Innosuisse - Swiss Innovation Agency under project number 19328.1 PFES-ES.

The paper presents results from a two-year long, federally funded (Innosuisse) project. As a proof-of-concept, the project implements a demonstrator based on the emerging recommendations of the Fairhair Alliance [2]. Low power nodes in a Thread network shall be provided with a secure bootstrapping process to be easily provisioned into an existing IT domain. The use of smartphones supports and simplifies this provisioning process. The public-key-based mutual authentication takes place between the low power nodes on one side and a Manufacturer Authorized Signing Authority (MASA) operated by the node manufacturer and a so-called Registrar operated by the building operator on the other side. As a result, the node receives an Operational Device Certificate and can legitimately join the IT domain. The paper illustrates the challenges encountered and proposes appropriate approaches

This paper is structured accordingly. We begin by describing the application environment. This includes a brief overview of the Thread protocol stack as well as a description of the challenge to establish trust between a Leaf Node and a building operator. We continue by outlining the proposed approach that we have implemented to establish such trust. We call this enrollment approach Autonomic Secure Bootstrapping. In the fourth section, we present the implementation of our Leaf Node followed by an overview of the test set-up in section five. We end with appropriate conclusions.

II. Application Environment

The following section uses an exemplary system from a building automation environment to illustrate the challenges. However, the described procedures and protocols can be readily adapted to other system environments.

A. Building Automation System Environment

Fig. 1 shows a system setup as typically found in a building automation environment. A Thread mesh network facilitates end-to-end IPv6 connectivity. The depicted network handles both, local communication between low power field nodes as well as IP based connectivity to upper data processing entities such as automation stations or cloud services. With regard to the building under control, these data processing entities can be located on premise or off premise.

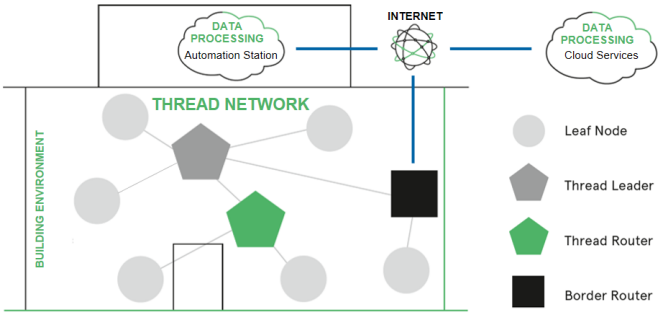

Fig. 1. Example of a building automation system based on a Thread mesh network

The displayed Thread network features nodes with distinct roles. Typically, there are Thread Routers, which are always on and route traffic among the devices. The routers form the mesh topology. One of the routers becomes the Thread Leader, which manages network parameters and coordinates the commissioning. In contrast to routers, there are Leaf Nodes. These are sleepy end devices specifically designed for low power operation. Such Leaf Nodes are often characterized by long sleep phases, resulting in high access latencies. As an example, a Leaf Node may periodically wake up from its sleep phase, take a measurement on its sensor and upload the result through the network to a data processing entity. To do so, it will make use of a Border Router, which interconnects the low power Thread network with a backbone network based on Wi-Fi or Ethernet. Remarkably, the Border Router does not employ a protocol translation like in a typical gateway device, but rather forwards the IPv6 packets natively. As a result, the Thread network behaves as a transparent IPv6 extension to the upper network.

B. Protocol stack

The Thread Group fosters and promotes an interoperable protocol stack based on different existing standards. Fig. 2 depicts a protocol stack on a Leaf Node in a Thread network.

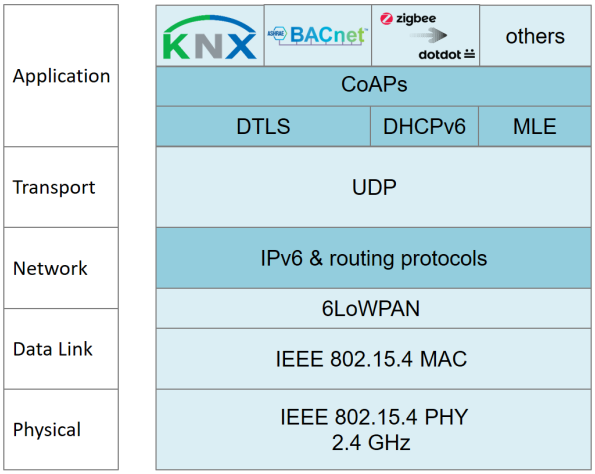

Fig. 2. Thread protocol stack on Leaf Node

Starting with the physical and data link layer, the stack is based on an IEEE 802.15.4 radio operating in the 2.4 GHz frequency band. The 6LoWPAN (IPv6 over Low-Power Wireless Personal Area Networks) adaption layer of IETF enables the use of IPv6 protocol and its powerful routing mechanisms on constrained devices. Above that, the connectionless UDP (User Datagram Protocol) forms the transport layer as it requires much less processor resources than the connection-oriented TCP (Transport Control Protocol). Meanwhile on the application layer, MLE (Mesh Link Establishment) configures and organizes the wireless mesh. DTLS (Datagram Transport Layer Security) provides end-to-end security in preventing eavesdropping, tampering, or message forgery. On top of DTLS, the RESTful web transfer protocol CoAPs (Secure Constrained Application Protocol) facilitates access to resources under URLs (Uniform Resource Locators). Specifically, such data transfers may include accesses among nodes within the Thread network itself as well as accesses between Thread network nodes and hosts outside of the Thread network. Particularly, the latter can e.g. be an automation station or a cloud service entity on the global internet.

At the top of the stack, the Leaf Node supports one or more application layer protocols to ensure semantic interoperability between devices. Possible specifications to organize the resources on a Leaf Node in a uniform way include KNX [3], BACnet [4] and Zigbee DotDot [5].

C. *Establishing Trust: From Supply Chain to System Integration*

A typical Leaf Node of a building automation system passes through many hands and many organizations before it is successfully and securely integrated into the IT network of a building operator. Fig. 3 shows the route of such a Leaf Node starting from its manufacturing site all the way to an accomplished system integration. Generally, a manufacturer mass-produces such Leaf Nodes without any customization or configuration for the customer site where the Leaf Node will be installed. Actually, in most cases the customer and the customer site are unknown at the time of production. Instead, the Leaf Node travels through a long supply chain with multiple organizations before it is installed at a building site. Frequently, contractors or personnel with a low trustfulness carry out the installation of the Leaf Node. As a result, the Leaf Node is exposed to many opportunities for malicious manipulations. Such potential manipulations include unauthorized replication, compromised firmware updates and deceiving reuse of device identities.

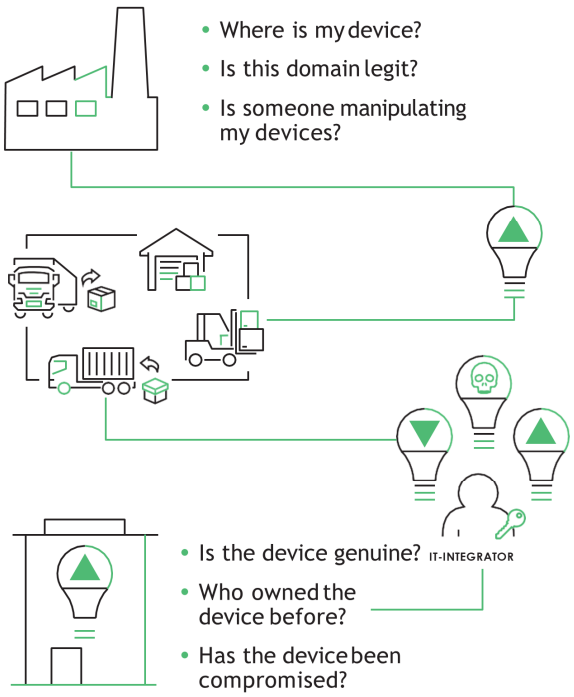

Fig. 3. The route of a Leaf Node (device) for a building automation system: From manufacturing site all the way to system integration

In a conventional process, eventually, an IT-Integrator or network administrator will have to enroll the installed Leaf Node manually into the specific IT environment of the building. Specifically, the IT-Integrator will have to decide, whether he trusts this particular Leaf Node. Accordingly, he will request solid proof that the Leaf Node is genuine, that it comes from a trusted manufacturer and so far has only been enrolled in environments by trustworthy previous owners. To be more specific, the IT-Integrator will manually enroll the Leaf Node into a domain. Likewise, this enrollment requires, that the Leaf Node trusts the IT-Integrator. A domain can be an IT environment or an application area like e.g. a heating system. In this context, a domain is a set of entities that share a common, organization-specific trust anchor. This trust anchor, under the responsibility of the network administrator, acts as a Domain Certification Authority (Domain CA).

However, as the number of nodes in building automation networks is growing rapidly, there is a strong need for an automatic enrollment process. Particularly, even untrustworthy personnel shall be able to trigger such an automatic enrollment. Still, the authentication and enrollment shall be secure and reliable. The following sections will describe such an approach. One key aspect is the reinforcement of the trust relationship between the manufacturer and the operator of the Leaf Node.

III. AUTONOMIC SECURE BOOTSTRAPPING

The following section outlines the key concepts for an automated enrollment in a building automation system. The described concepts are part of the emerging recommendations of the Fairhair Alliance, which again adopt concepts from IETF ANIMA working group [6]. Namely, these are described as "Bootstrapping Remote Secure Key Infrastructure (BRSKI)" [7] and "Enrollment over secure Transport (EST)" [8].

In the following, the term public-key certificate shall mean an electronic document used to prove the ownership of a public key. It includes information on the identity of the holder (e.g. the serial number) and is digitally signed by a Certificate Authority (CA). The public-key certificates are formatted according to X.509 [9] and [10]. In some cases, the text uses specific names, like e.g. Operational Device Certificate, for public-key certificates to indicate their context more precisely.

A. The Role of the Registrar

The proposed approach introduces a new entity, the Registrar. The objective is to replace the individual IT-Integrator (the person) with a fully automated service entity. The Registrar represents the individual domain of a building site and acts as a registration authority. Consequently, the Registrar decides whether a Leaf Node is allowed to join the domain. In case of a positive decision, the Registrar will forward an Operational Device Certificate issued by the Domain Certification Authority (Domain CA). The Operational Device Certificate is digitally signed by the Domain CA. It certifies the identity and ownership of the included public key. As a result, other members of the domain can use this public key to authenticate the Leaf Node as a legit member of the domain and to authorize access to services appropriately, i.e. for authorization of application-to application communication.

B. Initial Trust Relationships

As a prerequisite for the autonomic on-site enrollment process, two essential trust relationships have to be established prior to the start of the enrollment. Both, the individual new Leaf Node as well as the Registrar have to trust the Manufacturer Authorized Signing Authority (MASA). In this context, trust means that both instances can unequivocally authenticate the MASA and believe that the statements of the MASA are true. However, the new Leaf Node and the Registrar do not have any trust relationship with each other yet. Such trust will be established during the enrollment process. Fig. 4 illustrates these initial trust relationships.

To establish the trust of the Registrar in the MASA, many approaches are possible. For example, an IT-Administrator could configure this manually at a single point in time based on his trust in the manufacturer. Interestingly, for the described process the MASA does not have to trust the Registrar. Although the MASA may be able to authenticate the Registrar, it does not have to believe that statements issued by the Registrar are true.

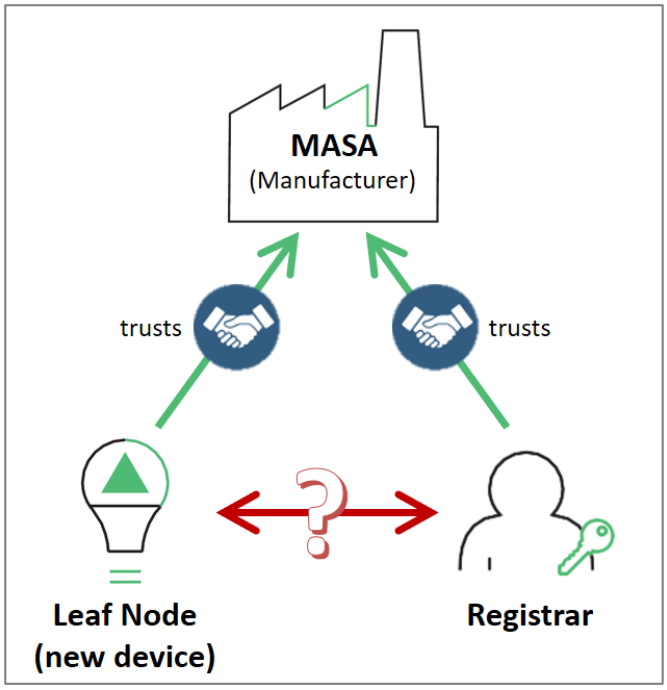

Fig. 4. Initial trust relationships before start of autonomic secure bootstrapping

Typically, the trust relationship of the Leaf Node to the MASA is imprinted at the manufacturing site. During the manufacturing process an Initial Device Identifier (IDevID, [11]) is created for each individual Leaf Node device. The IDevID is a device identifier that is cryptographically bound to the device and is securely stored inside the Leaf Node. Specifically, the IDevID includes a Manufacturer Device Certificate that contains an individual device serial number for the Leaf Node. Furthermore, the IDevID contains the manufacturer CA's public-key certificate, identifying the manufacturer CA as the root of trust.

C. Procedure Overview

Fig. 5 depicts the system components and their connections with regard to the Autonomic Secure Bootstrapping procedure. The figure extends the previously introduced example of a building automation system.

The process starts with the commissioning of the Leaf Node into the Thread network. The Leaf Node joins the on-site Thread network and gains IPv6 access. However, the Leaf Node does not have access to any applications yet as it yet lacks an operational identity. The Thread commissioning involves a straightforward application of the procedures described in [12]. Therefore, it will not be further detailed here.

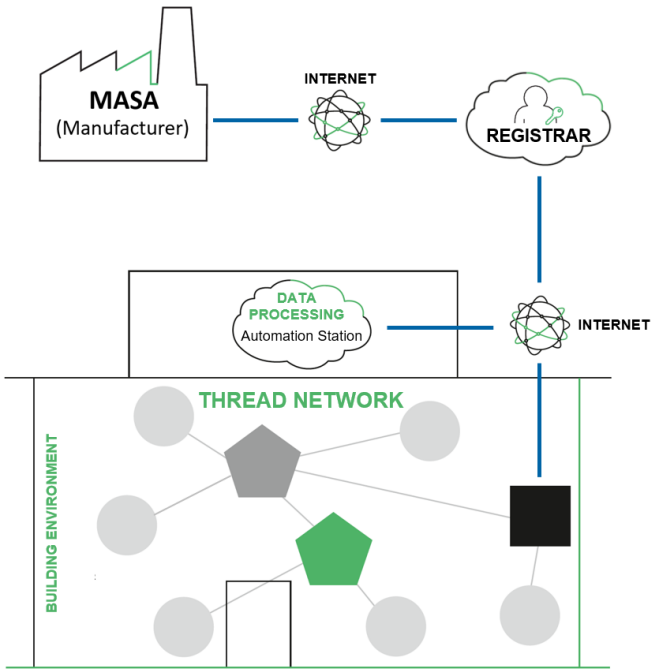

Fig. 5. System overview for on-site enrollment procedure

Once the Leaf Node has gained IPv6 access, the presented approach includes three steps to enroll a factory-new Leaf Node into a specific IT-environment of a building site.

1. *Establishing mutual trust between Leaf Node and Registrar*
   In this step, the Leaf Node and the Registrar mutually authenticate each other. Both trust the MASA. Therefore, the MASA facilitates the mutual authentication process with a Voucher [13]. After completion of this step, the Leaf Node has securely stored the Domain CA Certificate of the Registrar for future authentications. I.e., the Registrar has imprinted the factory-new Leaf Node with the appropriate key material of the domain. With the mutual trust in place, the Registrar can now optionally configure further trust relationships on the Leaf Node, e.g. the trust relationships to other trust anchors.
2. *Enrollment over secure transport (EST)*
   The Leaf Node wants to enroll itself into the domain. Therefore, the Leaf Node generates a new private/public key pair for which it requests an Operational Device Certificate from the Registrar, i.e. a public-key certificate issued by the Domain CA. The Operational Device Certificate is part of a locally significant secure device identifier (LDevID) [11]. It certifies the ownership of the private/public key pair and represents the identity of the Leaf Node. Therefore, at the end of this step the operational identity of the Leaf Node has been provisioned.
3. *Operational network enrollment*
   The Leaf Node uses the new Operational Device Certificate to authenticate itself within the domain. Based on this authentication, providers of application level services may grant the Leaf Node access to their services. In the chosen example, the Leaf Node can send data to the data processing entities, i.e. either the automation station or the cloud services.

The following sections describe these three individual steps in detail.

D. *Establishing mutual trust between Leaf Node and Registrar*

The factory-new Leaf Node discovers a Registrar and provisionally trusts it. There are many possible ways, how this discovery can take place. For example, it could be a pre-configured service record in DNS or a lookup in a resource directory using CoAPs.

By going through the Registrar, the Leaf Node requests the MASA to issue a Voucher. The Registrar relays this Voucher to the Leaf Node. Fig. 6 shows this sequence.

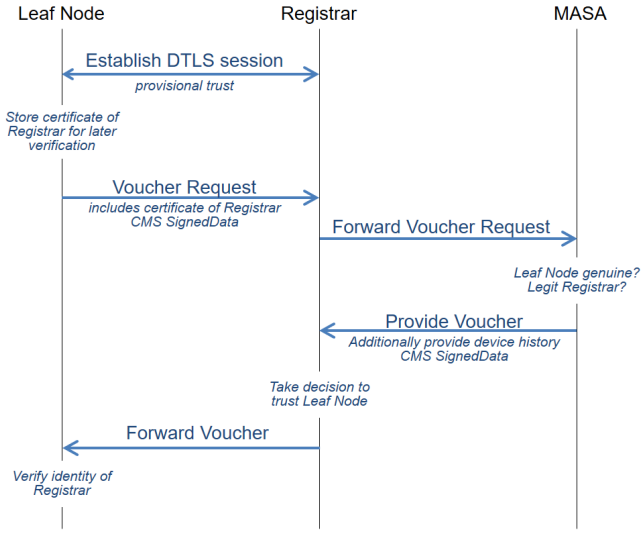

Fig. 6. Obtaining a Voucher from the MASA

The Leaf Node (via the Border router) initiates a DTLS handshake with the Registrar. After successful completion of the handshake, the Registrar and the Leaf Node provisionally trust each other and can communicate with each other through a secure channel. In the process of the DTLS handshake, the Leaf Node provides its Manufacturer Device Certificate to the Registrar. Optionally, the Registrar may choose to verify this

public-key certificate with a CA certificate obtained from the manufacturer. Similarly, the Registrar provides its Device Certificate to the Leaf Node. However, the Leaf Node cannot validate the Registrar yet. Therefore, the Leaf Node stores the Device Certificate of the Registrar for later verification.

In the next step, the Leaf Node submits a Voucher Request to the Registrar. It does so, through a CoAPs POST to a defined resource on the Registrar. The Voucher Request is a JSON-encoded object and contains

- A nonce, i.e. a newly created random number. At the reception of a Voucher, the nonce will allow to test for its "freshness".
- A serial number as a string. The serial number serves as a unique identification of the Leaf Node.
- The Device Certificate of the Registrar (signed by the Domain CA).

The Voucher Request is digitally signed and structured as a *SignedData* type of the Cryptographic Message Syntax (CMS) [14][1]. The signature ensures the integrity of the request. In addition, it identifies the specific Leaf Node as the issuer of the Voucher Request, i.e., as a legit product of a particular manufacturer.

In a further step, the Registrar forwards the Voucher Request to the MASA using a secured connection. For this purpose, the Registrar encapsulates the Voucher Request of the Leaf Node within a Registrar Voucher Request [7]. The Registrar duly signs the Registrar Voucher Request.

On reception of the Voucher Request, the MASA verifies that the request originates from a genuine Leaf Node of the manufacturer. This is achieved through verification of the CMS signature. At the minimum, the MASA logs the provisioning of the Leaf Node in an unforgeable device history, which is specific for the individual Leaf Node. Furthermore, it compares the Registrar identity included in the Voucher Request of the Leaf Node with the identity that the Registrar used to sign its Registrar Voucher Request to the MASA. The MASA confirms that it has indeed received the Voucher Request from the same Registrar that the Leaf Node has provisionally trusted. Therefore, even if this Registrar is not legit, the identity of this Registrar is now permanently logged in the device history of the Leaf Node. If at a later time a legit Registrar wants to enroll the same Leaf Node, it will see from the provided device history that this specific Leaf Node has been enrolled by an illicit Registrar. As a result, the legit Registrar may deny enrollment of the Leaf Node. Notably, the MASA does not take that decision, but through the device history, it transparently provides the information to a potential Registrar. At the end, it is always at the discretion of the Registrar, whether it wants to trust the Leaf Node or not.

Optionally, the MASA can apply further criteria before accepting the Voucher Request. E.g., that the Registrar indeed belongs to a domain in which the Leaf Node can be legitimately used. Eventually, the MASA issues a Voucher and sends it to the Registrar. In addition to the Voucher, it provides the Registrar with the device history of the Leaf Node. The issued Voucher is a CMS *SignedData* structure, signed with the MASA's private key. Particularly the Voucher contains the following objects:

- Nonce that matches the nonce in the Voucher Request.
- Assertion: A statement from the MASA whether it has verified the identity of the Registrar or alternatively has just logged it as the owner without verification.
- Domain CA Certificate (as provided by the Registrar in its Voucher Request)
- Serial number of the Leaf Node for which the Voucher has been created.

Using the Voucher and the device history, the Registrar can take a decision to trust the Leaf Node permanently. In addition, it may use a mixture of blacklists and whitelists for this decision. In case of a positive decision, the Registrar forwards the Voucher to the Leaf Node.

Eventually, the Leaf Node verifies, that the Voucher has been signed by the MASA. It does so with the use of the preinstalled MASA CA public-key certificate. Furthermore, it verifies the nonce and the serial number in the received Voucher. Additionally, it verifies the previously stored Device Certificate of the Registrar. It does so with the Domain CA Certificate that it received in the Voucher. In case all these verifications are successful, the Leaf Node accepts the Registrar and henceforth has permanent trust relationship with the Registrar. Importantly, the Leaf Node now holds the Domain CA public-key certificate. I.e., the Leaf Node has adopted the Registrar (on behalf of the Domain CA) as trustworthy. Among other things, the Leaf Node can use the public key from the Domain CA Certificate to verify that Operational Device Certificates presented by other devices in the domain have been duly signed by the Domain CA.

E. Enrollment over secure transport (EST)

Now that the Leaf Node and the Registrar have established mutual trust, the Leaf Node enrolls itself to the Domain. For this purpose, the Leaf Node has to obtain its individual Operational Device Certificate.

[1] similar to the *SignedData* type in PKCS#7 [24]

To begin with, the Leaf Node generates a new key pair, consisting of a public and a private key. Based on this key pair, the Leaf Node then generates a Certificate Signing Request (CSR) in PKCS#10 syntax [15]. The Leaf Node then sends the CSR to the Registrar, which provides an Operational Device Certificate from the Domain CA. The Registrar communicates the newly generated public-key certificate to the Leaf Node in a CSR response. Fig. 7 illustrates this process according to EST-coaps [16].

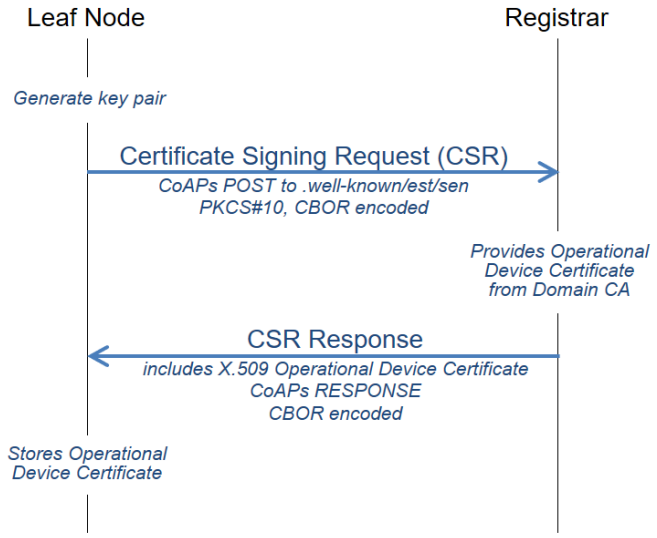

Fig. 7. Enrollment over secure CoAP (EST-coaps)

The Leaf Node uses a CoAPs request to send the CSR to a defined resource on the CoAPs server of the Registrar. The data serialization format of the CSR as well as of the CSR Response is Concise Binary Object Representation (CBOR) [17]. The CSR Response includes a public-key certificate (the Operational Device Certificate of the Leaf Node). At the completion of this step, the Leaf Node holds its individual Operational Device Certificate. It uses this to identify itself towards other devices in the domain.

F. *Operational network enrollment*

The Leaf Node can now finally authenticate itself on the application level.

IV. Implementation of Leaf Node

This chapter briefly summarizes the hardware and firmware of the Leaf Node. Furthermore, it introduces the concept of secure elements.

A. *Hardware*

The hardware design includes a Nordic nRF52840 System-on-Chip with 2.4 GHz PCB antenna. Furthermore, the design features the following sensors:

- InvenSense ICS 41350 microphone
- Bosch BME680 temperature, humidity, pressure and gas sensor
- Texas Instruments OPT3001 light sensor
- STMicroelectronics LSM6DSL accelerometer and gyroscope

Fig. 8 shows the printed circuit board of the Leaf Node. Once as a miniaturized break-out board on the left and once as a larger board with more connectors and extension opportunities on the right.

Fig. 8. Hardware implementation of Leaf Node

The Leaf Node achieves a calculated battery lifetime of 10 years on two AA battery cells. The figure is extrapolated from detailed power measurements and is based on a realistic choice of configuration parameters.

B. *Firmware*

The firmware is based on OpenThread (2018-09-26) [18], i.e. an open source implementation of the Thread protocol. This incorporates the driver for the Nordic SoC as well as the open source package mbedTLS [19] for the cryptographic applications. A dedicated own application layer implements the autonomic secure bootstrapping process introduced in this paper. Our application uses the cryptographic operations of mbedTLS. However, in this case we have a security threat, as the private keys are stored in debuggable memory. For this reason, the cryptographic operations can be transferred to an external device called a secure element.

C. *Secure Elements*

Secure elements provide hardware accelerated support for cryptographic operations and tamper proof memory for the secure storage of cryptographically sensitive material. Moreover, they employ specific techniques against so-called side channel attacks. Secure elements physically isolate private keys and other secret cryptographic material from the application. Therefore, they significantly reduce the exposure to cyber security attacks. Typically, they use an I2C interface to

communicate with the host microcontroller. Private keys can be generated inside the protected environment of the secure element and never have to leave the secure element. The secure element takes care of all cryptographic operations involving private keys. Furthermore, secure elements can store public-key certificates as a root of trust, ensuring that this root of trust can only be changed by authorized parties.

The Leaf Node features secure elements from four different vendors. Namely, these are Microchip, NXP, Infineon and Trusted Objects. With all of them, we have made extensive power as well as speed measurements. However, the development effort and expertise required to integrate the individual secure elements turned out to be extremely high. If secure elements are to be widely adopted and deployed in IoT devices, a major harmonization of the interfaces is required.

V. VERIFICATION

This section briefly describes the verification of the Leaf Node. Clearly, the focus of the verification is on showing that the described Leaf Node with its constrained resources can successfully perform all the required actions for the Autonomic Secure Bootstrapping. However, the detailed test sequences and the achieved results are outside the scope of this paper.

A. Test Set-up

Fig. 9 gives an overview of the test set-up. As the Leaf Node is the Device Under Test (DUT), only a limited set of functionalities has been implemented for the entities Registrar, Data Processing and MASA.

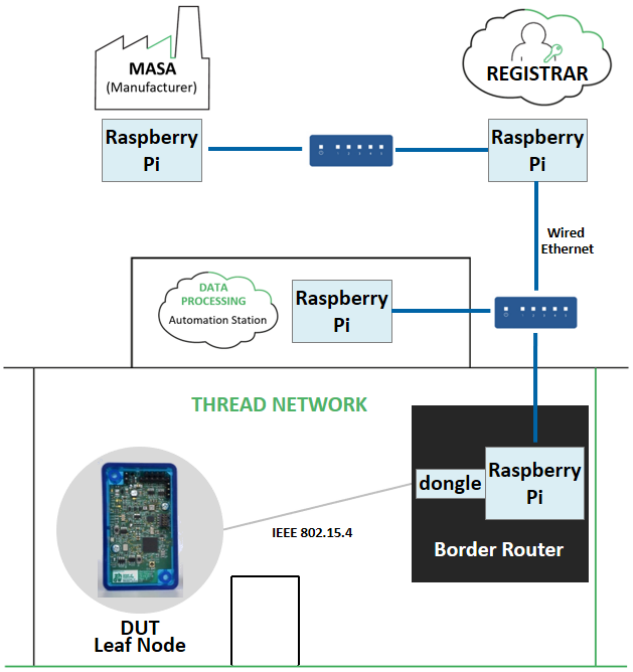

Fig. 9. Test set-up

At power-up, the Leaf Node initiates the Autonomic Secure Bootstrapping process. It establishes mutual trust with the Registrar, enrolls itself in the domain and finally authenticates itself to the Data Processing entity and pushes data to it.

In addition to the test described here, further integration and system testing have been performed. These tests use a fully functional implementation of a Registrar in the cloud that is currently participating in plug-fest tests for EST-coaps interoperability.

B. Border Router

The Border Router consists of a Raspberry Pi Version 3b with an nRF52840 dongle [20] connected on the USB port. The dongle runs OpenThread in Network Co-Processor (NCP) mode [21]. Likewise, the Raspberry runs the OpenThread Border Router software [22]. As a result, the Border Router routes IPv6 traffic between the wireless Thread network devices on the local network and the external IP network connected through wired Ethernet.

C. Registrar, Data Processing and MASA

The applications on these instances use Californium [23], a framework for CoAP targeted at back-end services. As part of Californium, Scandium is used for DTLS.

VI. CONCLUSIONS

This paper presents the key concepts of an Autonomic Secure Bootstrapping process for resource-constrained devices in building automation applications. The described security concepts are based on the emerging standards of the Fairhair Alliance. Manual enrollment of IPv6 accessible field devices is replaced with a fully autonomic and automated enrollment process. The entire process has been successfully implemented and tested on our own Leaf Nodes in a Thread network. Specifically, the project team gained hands-on experience with secure elements of four different Silicon vendors. Although it is possible to implement the described process with commercially available secure elements, the required development effort and expertise to integrate such secure elements turned out to be prohibitively high. The authors see a strong need to harmonize the use of secure elements. Especially on the firmware side, integration of secure elements has to be made tremendously simpler. This will be a mandatory prerequisite to ensure security in low power IoT nodes and therefore to enable their roll-out in huge numbers.

VII. REFERENCES

[1] Thread Group, Inc., "Thread," [Online]. Available: https://www.threadgroup.org/. [Accessed 19 10 2018].

[2] Fairhair Alliance, "The Fairhair Alliance," [Online]. Available: https://www.fairhair-alliance.org/. [Accessed 19 10 2018].

[3] KNX Association, "KNX - Smart home and building solutions," [Online]. Available: https://www.knx.org. [Accessed 23 10 2018].

[4] ASHRAE SSPC 135, "BACnet/IP," [Online]. Available: http://www.bacnet.org/Tutorial/BACnetIP/index.html. [Accessed 23 10 2018].

[5] Zigbee Alliance, "Dotdot," [Online]. Available: https://www.zigbee.org/zigbee-for-developers/dotdot/. [Accessed 23 10 2018].

[6] IETF Anima WG, "Autonomic Networking Integrated Model and Approach (anima)," [Online]. Available: https://datatracker.ietf.org/wg/anima/about/. [Accessed 19 10 2018].

[7] IETF Anima WG, Internet-Draft: Bootstrapping Remote Secure Key Infrastructures (BRSKI), Vols. draft-ietf-anima-bootstrapping-keyinfra-16, 2018.

[8] Internet Engineering Task Force (IETF), RFC 7030: Enrollment over Secure Transport, 2013.

[9] ITU-T - TELECOMMUNICATION STANDARDIZATION SECTOR OF ITU, "ITU-T Recommendation X.509: Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks," 2016.

[10] Internet Engineering Task Force (IETF), RFC 5280: Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, 2008.

[11] MAN, LAN / Committee, Standards, IEEE Std 802.1AR-2018, IEEE Standard for Local and Metropolitan Area Networks—Secure Device Identity, New York: IEEE Computer Society, 2018.

[12] Thread Group, Thread Specification, Revision 1.1.1, 2017.

[13] Internet Engineering Task Force (IETF), RFC 8366: A Voucher Artifact for Bootstrapping Protocols, 2018.

[14] Internet Engineering Task Force (IETF), RFC 5652: Cryptographic Message Syntax (CMS), 2009.

[15] Internet Engineering Task Force (IETF), RFC 2986: PKCS #10: Certification Request Syntax Specification, 2000.

[16] Internet Engineering Task Force (IETF), EST over secure CoAP (EST-coaps) draft-ietf-ace-coap-est-06, 2018.

[17] Internet Engineering Task Force (IETF), RFC 7049: Concise Binary Object Representation (CBOR), 2013.

[18] OpenThread Nest Labs, "OpenThread," [Online]. Available: https://openthread.io/. [Accessed 27 10 2018].

[19] ARM Limited, "SSL Library mbed TLS / PolarSSL," [Online]. Available: https://tls.mbed.org/. [Accessed 23 10 2018].

[20] Nordic Semiconductor, "nRF52840 Dongle," [Online]. Available: https://www.nordicsemi.com/eng/Products/nRF52840-Dongle. [Accessed 28 10 2018].

[21] OpenThread Nest Labs, "Network Co-Processor Support | OpenThread," [Online]. Available: https://openthread.io/guides/ncp/. [Accessed 28 10 2018].

[22] OpenThread Nest Labs, "OpenThread Border Router | OpenThread," [Online]. Available: https://openthread.io/guides/border-router. [Accessed 28 10 2018].

[23] The Eclipse Foundation, "Californium (Cf) CoAP framework," [Online]. Available: https://www.eclipse.org/californium/. [Accessed 29 10 2018].

[24] Internet Engineering Task Force (IETF), RFC 2315: PKCS #7: Cryptographic Message Syntax, 1998.