Published in IEEE journal on selected areas in communications, 2015, vol. 33, no. 4, pp. 690-702 which should be cited to refer to this work.

Towards a Lightweight Authentication and Authorization Framework for Smart Objects

José L. Hernández-Ramos*, Marcin P. Pawlowski§†, Antonio J. Jara†, Antonio F. Skarmeta* and Latif Ladid‡
*Department of Information and Communications Engineering, Computer Science Faculty, University of Murcia
Murcia, Spain
{jluis.hernandez, skarmeta}@um.es
§Department of Information Technologies, Faculty of Physics, Astronomy and Applied Computer Science, Jagiellonian University
Krakow, Poland
{marcin.pawlowski@uj.edu.pl}
†Institute of Information Systems, University of Applied Sciences Western Switzerland (HES-SO)
Sierre, Switzerland
{jara@ieee.org}
‡IPv6 Forum and University of Luxembourg
Luxembourg
{latif@ladid.lu}

***Abstract*—The Internet of Things (IoT) represents the current technology revolution that is intended to transform the current environment into a more pervasive and ubiquitous world. In this emerging ecosystem, the application of standard security technologies has to cope with the inherent nature of constrained physical devices, which are seamlessly integrated into the Internet infrastructure. This work proposes a set of lightweight authentication and authorization mechanisms in order to support smart objects during their life cycle. Furthermore, such mechanisms are framed within a proposed security framework, which is compliant with the *Architectural Reference Model* (ARM), recently presented by the EU FP7 IoT-A project. The resulting architecture is intended to provide a holistic security approach to be leveraged in the design of novel and lightweight security protocols for IoT constrained environments.**

I. Introduction

Recent advances in wireless communications and pervasive computing are driving the constant development of the so-called Internet of Things (IoT) [1], providing ubiquity and intelligence to our surrounding environment. IoT represents the extension of *Information Technology* (IT) to all areas of our lives, transforming current isolated networks and infrastructures into a global network of interconnected heterogeneous objects as a key enabler of the future Big Data era [2]. In fact, the increasing interest on IoT from academia and industry is promoting the emergence of innovative services to be leveraged by societies, and enabling unprecedented economic and social opportunities for governmental and private organizations in the envisioned Smart Cities ecosystems [3].

Over recent years, significant technological challenges have been solved through the extension and adaptation of wireless communication technologies and protocols. In particular, several IETF working groups, such as *IPv6 over Low power WPAN* (6LoWPAN) and *Constrained RESTful Environments* (CoRE), are focused on the adaptation of existing Internet protocols to more efficient, interoperable and lightweight versions to be used on constrained environments. These protocols are intended to enable a seamless inclusion of smart objects into the Internet to realize the scenarios which are envisaged by IoT community. Specifically, the main goal of the 6LoWPAN WG is the adaptation of the IPv6 protocol to be employed on constrained environments such as IEEE 802.15.4 networks, in order to obtain end-to-end connectivity between constrained devices and any entity connected to the Internet [4], [5]. These adjustments are based on header compression and encapsulation mechanisms. Moreover, the CoRE WG was specifically founded to define an application layer protocol for resource constrained devices. As a result, the *Constrained Application Protocol* (CoAP) [6] was designed. This protocol, based on the same RESTful principles as HTTP, allows the realization of embedded services but accommodated to the requirements of constrained devices and networks [7].

In spite of such remarkable efforts, the application of security mechanisms to be deployed on this new generation of pervasive scenarios still remains as the main concern for a global IoT deployment [8]–[10]. In fact, the realization of these scenarios requires to address significant security and access control implications, since physical and constrained devices are being seamlessly integrated into the Internet infrastructure with network and processing abilities, making them vulnerable to attacks and abuse [11]. However, current security and access control solutions were not designed with these aspects in mind and they are not able to meet the needs of these incipient ecosystems regarding scalability, interoperability, lightness and end-to-end security [12], [13]. In this direction, the IETF *Authentication and Authorization for Constrained Environments* (ACE) WG has been recently established to produce a standardized security solution to be

used by devices and networks with tight resource constraints. Specifically, the work of ACE WG is focused on the design and development of authentication and authorization mechanisms to enable authorized access to resources, which are hosted in constrained smart objects.

Under the main foundations of ACE WG, this work provides a set of lightweight authentication and authorization mechanisms, as well as their application on IoT constrained environments. These mechanisms are integrated and extended with other standard security technologies in order to support smart objects during its life cycle. In particular, in this work we consider the use of an lightweight version of *Extensible Authentication Protocol over LAN* (EAPOL) [14] to initiate a security bootstrapping process by integrating standard technologies, such as EAP [15] and *Remote Authentication Dial In User Service* (RADIUS) [16]. This process has been extended with *Extensible Access Control Markup Language* (XACML)-based authorization procedures [17] for obtaining lightweight access tokens [18] to be employed at operational plane achieving end-to-end secure communication between constrained devices. Furthermore, such mechanisms are framed within a security framework which is compliant with the *Architectural Reference Model* (ARM) [19], recently presented by the EU FP7 IoT-A initiative [1]. While this work is focused on authentication and authorization mechanisms, the proposed framework is intended to provide a holistic security approach to be leveraged by IoT devices throughout their life cycle. Additionally, a set of evaluation results is analysed and discussed to demonstrate the suitability of the proposed mechanisms.

The remainder of this paper is structured as follows. Section II analyses recent proposals addressing different security aspects on IoT scenarios. Section III presents our ARM-compliant framework for security management of IoT devices during their life cycle. Section IV gives a description of the proposed lightweight security mechanisms, whose integration is provided in Section V. In Section VI, we present several experimental results of the proposed mechanisms, while the integration of the proposed approach into the emerging IETF ACE WG is discussed in Section VII. Finally, in Section VIII, we end up with some conclusions and an outlook of our future work in this area.

II. Related Work

The application of security mechanisms on IoT scenarios has to address new requirements due to the nature and tight constraints of devices and networks composing these incipient ecosystems. This has given rise to a broad consensus among academia and industry to consider security as the main barrier to be overcome in next years for a global deployment of IoT [20]–[23]. These challenges have attracted a huge attention from the research community, and recently several efforts are beginning to emerge addressing different security aspects during the life cycle of smart objects.

Regarding the application of security mechanisms at the bootstrapping stage for constrained devices, the authors in [24] provide an authentication and key establishment scheme for WSNs in distributed IoT applications. The proposal, which is also envisioned for bootstrapping phase, is based on a simplified *Datagram Transport Layer Security* (DTLS) [25] exchange and the use of TinyECC [26] for cryptographic operations. [27] provides the main bootstrapping approaches and protocols to be considered on IoT environments. Specifically, EAP [15] is established as the standard authentication framework for this process due to its maturity and flexibility. Additionally, three alternative protocols are analysed for security bootstrapping: *HIP Diet EXchange* (HIP-DEX) [28], *Protocol for Carrying Authentication for Network Access* (PANA) [29] and 802.1X [30]. The use of HIP-DEX for the network access stage is analysed in [31]. Although the results shown are promising compared to DTLS, HIP-DEX is not widely adopted. This is mainly because it does not provide native support for certificate-based public key agreement and the high complexity of the puzzle mechanism to mitigate DoS attacks. Moreover, the authors in [32] provide a lightweight implementation of PANA called *PANATIKI* to be deployed on constrained devices. Due to the high cost of public key cryptographic operations, it is based on the use of *Extensible Authentication Protocol-Pre-Shared Key* (EAP-PSK) [33] as the authentication method, providing a lower degree of scalability and security. In addition, previous proposals assume that the device has already been configured with an IP address prior the network access stage, which can stand for a potential security threat for the network. Alternatively, our approach operates below the network layer, by using 802.1X to transport EAP messages at the bootstrapping stage, offering a lightweight mechanism suitable to the requirements of resource-constrained environments.

At operational plane, CoAP [6] has been recently declared as a standard specialized web transfer protocol to be deployed on constrained devices and networks. CoAP offers several modes for securing the protocol through a security binding to DTLS [25], which requires a heavy message exchange to agree security parameters. Furthermore, it does not cover the use of authorization and access control mechanisms at the application level. In this direction, the work presented in [34] provides an approach based on *Elliptic Curve Cryptography* (ECC) for key establishment and *Role-Based Access Control* (RBAC) model [35] for the definition of access control policies. They consider an inter-domain scenario in which different registration authorities are responsible for the authentication process. Several security gaps of this work are discussed in [36], in which different enhancements are proposed in order to satisfy the basic security properties of the scenario. Moreover, an authentication and access control scheme for the layer perception of IoT is given in [37]. It is based on an efficient key establishment making use of ECC and *Attribute-Based Access Control* (ABAC) [38], which requires a complex management and hinders its application to constrained devices. Consequently, they only provide theoretical results of the proposed model. Recently, several access control mechanisms based on authorization tokens have been proposed on IoT scenarios [39], [40]. These approaches are based on the externalization of authorization decisions in a central entity which

[1] IoT-A: http://iot-a.eu

issues privileges to be enforced at the end device [41]. Under the main foundations of these works, *Distributed Capability-Based Access Control* (DCapBAC) [18] has been recently introduced as a feasible access control approach to be deployed on constrained environments. DCapBAC allows a distributed approach in which constrained devices are enabled with authorization logic by adapting the communication technologies and data-interchange format. Specifically, it makes use of *JavaScript Object Notation* (JSON) [42] as representation format for the token, the use of emerging communication protocols such as CoAP and 6LoWPAN, as well as a set of cryptographic optimizations for ECC [43].

While previous works address partially security concerns on IoT environments, this work presents an integral scenario for the security management of a constrained device during its life cycle. In particular, we provide a lightweight security bootstrapping process based on standard technologies for obtaining authorization tokens. These credentials are used by devices through DCapBAC at the operation stage enabling a secure *Thing-to-Thing* (T2T) communication.

III. ARM-compliant security framework for supporting the life cycle of IoT devices

The secure management of the whole life cycle of IoT devices is one of the key challenges to be tackled for a broad adoption of the Internet of Things. Nowadays, typical personal mobile devices such as smartphones or tablets provide efficient user interfaces which are used for management and maintenance tasks by people. However, the deployment of constrained IoT devices (e.g. sensors or actuators) on uncontrolled scenarios, such as smart cities, triggers new requirements to be overcome for an effective and secure management of such devices. Nowadays, management tasks are usually provided by manual maintenance and proprietary solutions which are tailored to a specific device or service. This lack of automated mechanisms leads to security breaches that can be potentially exploited by malicious entities during the whole life cycle of IoT devices.

Figure 1 shows the different phases of a smart object during its life cycle and the security levels which must be considered [12] [44]. The life cycle begins when an IoT device is installed and commissioned in a network during the bootstrapping process. This stage includes an authentication and access control process in order to provide cryptographic material and parameters, which can be used by the device for secure access to services. Furthermore, this stage can make use of security credentials which were provisioned to the device at the manufacturing process. Subsequently, the smart object starts to operate providing the corresponding services for which it was created (e.g. temperature values). During this stage, the consideration of security mechanisms is necessary to protect access to resources that are hosted on the device. Optionally, a smart object could be maintained in order to be upgraded, reconfigured, and consequently commissioned again. Finally, it can be decommissioned, in which case, the revocation of the corresponding security credentials that were acquired during its life cycle is required.

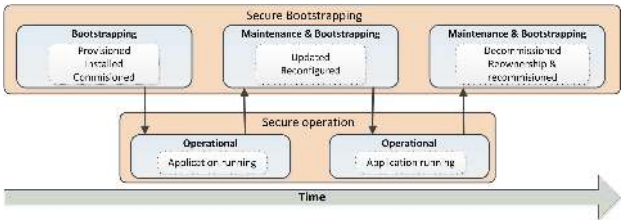

Figure 1: Security planes for the life cycle of a smart object

The high complexity for a secure management of smart objects throughout their whole life cycle imposes the need to consider architectural approaches, taking into account the inherent requirements of the application of security mechanisms and protocols on IoT scenarios. The huge range of application scenarios of IoT has led in recent years to the specification of different architectures which are usually tailored to be deployed on specific domains or addressing particular requirements. This has been identified as one of the main barriers for IoT adoption on a broad scale and the main incentive for the development of coordinated efforts driven by the *Internet of Things European Research Cluster* (IERC), in order to define a common and harmonized IoT architecture to be used by industry and academia. One of the first proposals to address this issue was IoT-i [45], an European research project which dealt with the analysis of different architectures to create a joint and aligned vision of the IoT in Europe. This effort meant a step forward in order to develop a holistic environment that encourages a broader adoption of IoT. IoT-A [19] was a large-scale project focused on the design of an *Architectural Reference Model* (ARM) to instantiate IoT architectures through a set of specific tools and guidelines. The main motivation of this reference architecture was to optimize the interoperability among isolated IoT applications to create a global ecosystem of services under a common understanding. This promoted additional initiatives adopting ARM as the starting point of design activities, favoring the alignment of architectures and enabling to reuse functionalities and components among different application domains. In this direction, the focus of the architecture proposed by IoT6 [46] is to use the results of previous projects to design an IPv6-based service-oriented architecture, in order to achieve a high degree of interoperability between different applications and communication technologies. Additional architectures were proposed by other remarkable efforts at European level, such as BUTLER [47], SENSEI [48] or FI-WARE [49] based on the specific set of requirements from particular application domains. On the one hand, SENSEI was focused on designing the service layer in wireless sensor and actuators networks. On the other hand, FI-WARE, under the FI-PPP program, designed an open platform based on an architecture to be leveraged by the so-called Future Internet. However, security concerns, which are critical in the design of innovative and valuable services to be deployed on IoT scenarios, are not the main focus of such architectures. To fill this gap, our ARM-compliant security framework [50] addresses these requirements by instantiating and extending the security functional group of ARM, which

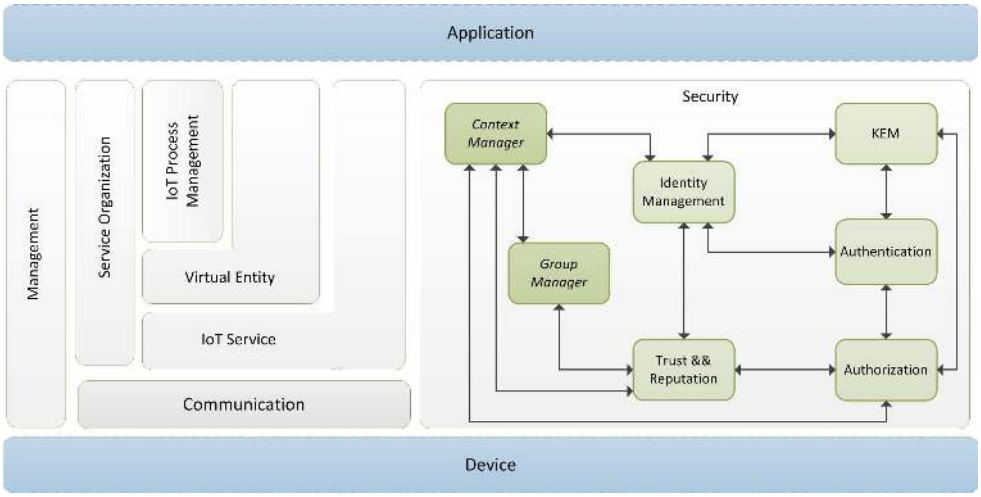

Figure 2: ARM-based Security Framework for the IoT

promotes its applicability and interoperability in a wide range of IoT scenarios in which security and privacy are required.

Figure 2 shows our ARM-compliant architecture, in which the security functional group is detailed. In addition to the functional components of ARM, we consider an extension of it, in order to address more flexible data sharing models in which some information can be shared with a group of entities or a set of unknown receivers and, consequently, not addressable a priori. Furthermore, due to the pervasive character of envisioned IoT scenarios, the way in which this information is disseminated must consider contextual data where sharing transaction is going to be performed. Therefore, the proposed framework is not just an instantiation of the security functional group of ARM, but it actually extends it by defining additional components which can be considered by emerging IoT scenarios. While the integration of this framework is being considered and analysed under several EU Research Projects, in this work we focus on the Authentication and Authorization functional components in order to provide suitable mechanisms addressing different security planes according to the life cycle of constrained smart objects.

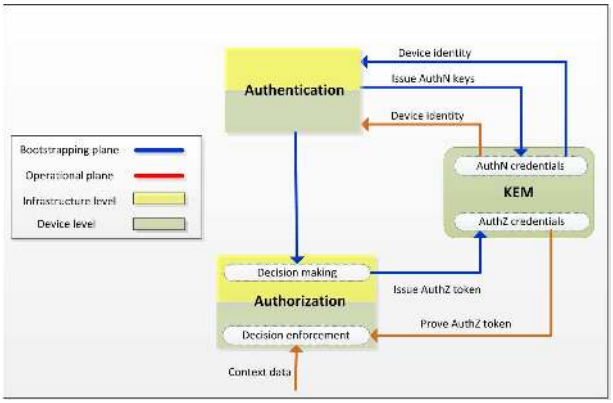

Figure 3: Functional components and interactions involved in the proposed scenario

In addition, Figure 3 shows the main functional components and interactions involved in our proposal. At bootstrapping plane, the Authentication functional component is responsible for authenticating the device by using the corresponding infrastructure (e.g. AAA). As a result of a successful authentication, a set of keys is derived (e.g. a *Master Session Key* (MSK) and an *Extended Master Session Key* (EMSK) in the case of EAP) and used to establish security associations between the device and the infrastructure components. Furthermore, our approach considers the extension of this process through an authorization mechanism by which an authorization credential is inferred and sent to the device through the infrastructure. The Authorization functional component is in charge of this process, while the credentials obtained in this phase are stored in the Key Exchange and Key Management (KEM) functional component. At operation plane, the same modules are needed at the device level. In particular, the Authentication component is responsible for checking that the requester device is who it claims to be. In addition, the previously obtained authorization credential is sent to the target device for authorization enforcement. This process takes into account context parameters which are locally detected by the target device and received from the Context Manager functional component. The instantiation of these components in network elements, as well as the required message exchange, are described in Section V.

IV. Lightweight Security Mechanisms for IoT Constrained Environments

The explanation of the proposed authentication and authorization mechanisms is split according to the different stages of a smart object during its life cycle. On the one hand, security considerations at bootstrapping level, as well as the proposed set of optimizations, are presented. On the other hand, the proposed security mechanisms at operational level are described. These mechanisms have been designed taking into account the severe resource constraints of current IoT devices and networks, as well as the initial set of considerations and requirements of the recent IETF ACE WG [51].

A. Security Bootstrapping

The bootstrapping process usually consists of a set of procedures in which a node is installed and commissioned within a network. Optionally, this stage can include authentication and access control mechanisms to get security parameters for trusted operation. For a successful and secure bootstrapping process, well-known mechanisms need to be on the basis. Additionally, in the context of IoT constrained scenarios, the application of such procedures need to be analysed due to implicit requirements of these environments.

Currently, EAP [15] is widely used and recognized as the standard mechanism to provide flexible authentication through different EAP methods. According to EAP terminology, these methods allow an *EAP peer* to be authenticated by an *EAP server* through *EAP authenticator* for network access. Moreover, these EAP methods can provide keying material after successful authentication. Depending on the place in where the EAP server is located, there are two possible configurations: *standalone* and *pass-through*. In the former option, the EAP Server is collocated with the EAP authenticator and the communication between the two components is local. In the pass-through configuration, the EAP Server is located on a different node and an additional AAA protocol is required for this communication (e.g. RADIUS). For communication between the EAP peer and the EAP authenticator, an additional protocol is needed to transport EAP messages. For this purpose, there are several options which operate at different communication layers, such as PANA [29], 802.1X [30] and *Internet Key Exchange* (IKE) [52]. The *Internet Protocol security* (IPsec) and IKE have been evaluated by [53] and [54], whose results show that both options could require excessive cost for constrained environments. Moreover, PANA operates on top of IP layer and nodes need to be addressable at the bootstrapping process before being authenticated. Moreover, the use of PANA requires more overhead and processing requirements than a solution operating at a lower level. Consequently, in this work the transport of EAP messages is considered at link layer.

	Frame Size	Overhead	Ratio
EAPOL in 802.11	2304 bytes	6 bytes	0.26%
EAPOL in 802.15.4	127 bytes	6 bytes	4.72%
SEAPOL in 802.15.4	127 bytes	3 bits	0.59%

Figure 4: Comparison of EAPOL and SEAPOL overhead in 802.11 and 802.15.4 frames

In particular, the IEEE 802.11i standard [14] introduced the EAPOL protocol for 802.11 wireless networks. The standard approach requires 6 bytes of a 802.11 frame, which represents a 0.26% of the frame size. However, in case of 802.15.4 networks, EAPOL represents almost 5% of the frame size. This creates the need to design a more lightweight and optimized solution for environments with tight resource constraints. The proposed approach has been designed by considering the main EAPOL functionality can be represented by only 5 different frame types. In addition, the *EAPOL Start* and *EAP Packet* frames could use the same frame type and be easily differentiated by the frame payload size. This gives the possibility to compress EAPOL in just 3 bits (93.75% less overhead in comparison to the regular EAPOL) as shown in Figure 4. The proposed has been called *Slim EAPOL* (SEAPOL) and the required modifications of 802.15.4 to support it are shown in Figure 5. Furthermore, it should be noted that SEAPOL does not require additional space because it makes use of 3 reserved bits of the IEEE 802.15.4 frame header, which are always sent (and unused) during data messages exchange.

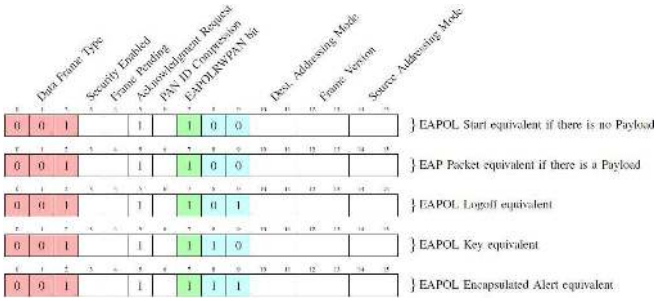

Figure 5: IEEE 802.15.4 Frame Control field modifications to support Slim Extensive Authentication Protocol over Low-Rate Wireless Personal Area Networks

Besides the use of SEAPOL to transport EAP messages between the EAP peer and the EAP authenticator, AAA infrastructures can be used for an authentication and access control process in which cryptographic material and configuration parameters can be obtained by the corresponding IoT device, enabling a secure operation. Section 5 provides a detailed description of the proposed scenario, in which SEAPOL and AAA infrastructures are extended to deliver authorization tokens to constrained devices.

B. Operational Security

At operational level, security guarantees that only trusted and legitimate instances of an application running in the IoT can communicate with each other, through the use of the corresponding security mechanisms at the application layer. Specifically, CoAP [6] defines a security binding to DTLS [25] through the use of pre-shared keys, raw public keys or certificates. However, it does not cover the use of authorization and access control mechanisms at the application level. Because of the strong constraints of IoT devices and networks, in recent years, the protection of resources and services which are provided by smart objects, has been mainly addressed by centralized architectures, in which a back-end server or gateway is responsible for security tasks. While traditional access control models and security standard technologies and protocols can be used in these approaches, several drawbacks arise when they are considered on a real deployment. On the one hand, the inclusion of a central entity prevents end-to-end security to be achieved. On the other hand, these solutions cannot provide a suitable level of scalability for smart environments with a potentially huge amount of constrained devices. Furthermore, due to the fact that a single entity stores and manages all the data from a set of devices, any

vulnerability might compromise a vast amount of sensitive information.

While most of mentioned drawbacks of centralized approaches can be solved, a distributed security approach requires the analysis and adaptation of the standard mechanisms to be deployed on devices with tight resource constraints. Recently, the *Distributed Capability-Based Access Control* model (DCapBAC) has been postulated as a realistic approach to be used on IoT scenarios [18]. This model offers several advantages over more established approaches and it is gaining attention from research community providing support for least privilege, and providing a greater level of suitability to cope with the inherent requirements of a distributed approach. The key element of DCapBAC is the capability. The concept of capability was originally introduced in [55] as *"token, ticket, or key that gives the possessor permission to access an entity or object in a computer system"*. This concept is applied to IoT environments and extended by defining conditions which are locally verified on the constrained device. This feature enhances the flexibility of DCapBAC since any parameter which is read by the smart object could be used in the authorization process. Moreover, it is based on technologies which have been proposed recently to be used in constrained environments. In particular, authorization tokens are specified with JSON [42], which are sent to the target device by using CoAP. Moreover, DCapBAC is based on the use of *Public Key Cryptography* (PKC), whose features fit IoT requirements regarding scalability and interoperability. Specifically, the application of PKC is based on a set of cryptographic optimizations for ECC for point and field arithmetic [56], which are implemented on the smart object in order to design a lightweight, end-to-end secure authorization process. Based on these optimizations, DCapBAC specifies a simplified DTLS-based exchange for mutual authentication and key establishment through *Elliptic Curve Diffie-Hellman Ephemeral* (ECDHE) and *Elliptic Curve Digital Signature Algorithm* (ECDSA) [57].

The security approach at operational plane is based on DCapBAC. However, while the proposed scenario in [18] is mainly intended for a *Human-to-Thing* (H2T) communication, this work is intended to achieve a transparent *Thing-to-Thing* (T2T) communication in order to provide the benefits of a decentralized approach for IoT scenarios in terms of end-to-end security, scalability, interoperability, and flexibility. In addition, this work addresses the token generation stage in order to provide an integral approach for the security management during the life cycle of resource-constrained devices.

V. Proposed scenario

The set of optimizations which were presented in the previous section has been used for the definition of an integral security approach for IoT constrained environments, which is shown in the Figure 6. This scenario illustrates the application of the proposed mechanisms during different stages of the life cycle of an IoT device. Additionally, they have been combined with other standard security technologies in order to realize the desired functionality. Before the explanation of the required message exchange for our approach, a description of the main components involved is provided. This specification has been defined based on the actors which are considered by the IETF ACE WG in [58] and the assumptions made in [51] for IoT constrained environments:

- **Client device (C)**. Resource-constrained device acting as a client to access services hosted on another constrained device. In addition, it acts as *Supplicant* (802.1X terminology) or *EAP peer* (EAP terminology) during the bootstrapping stage.
- **Server device (RS)**. Resource-constrained device acting as a resource server receiving access requests. Such requests are based on REST configuration by using CoAP methods (i.e. GET, POST, PUT and DELETE). Sensor values or configuration data are examples of resources.

Additionally, we consider both devices have valid certificates with an associated key-pair, which was given during the provisioning or manufacturing process. Moreover, it is assumed RS has no prior knowledge of C at the time of the access request.

- **EAP Authenticator**. Also known as AAA client, it is the entity (non-constrained) responsible for providing network access. In the proposed scenario, it is assumed that it operates on *pass-through* configuration, forwarding EAP packets between the AAA server (RADIUS) and the EAP peer.
- **EAP Server**. The component (non-constrained) that provides an authentication service to the authenticator by checking the credentials provided by the EAP peer. IT is based on RADIUS and has been extended to request authorization tokens to be sent to the EAP peer after a successful authentication.
- **Authorization Server (AS)**. It is the entity (non-constrained) in charge of authorization tasks. It consists of two subcomponents:
  - **Policy Decision Point (PDP)**. This entity evaluates authorization policies and makes authorization decisions. It is based on the XACML standard [17] and supports the *Multiple Decision Profile* (MDP) [59].
  - **Capability Manager (CapM)**. It is responsible for receiving requests from the EAP Server to generate authorization tokens for the EAP peer. Furthermore, it sends authorization requests to the PDP in order to obtain the set of privileges to be embedded into the token.

Moreover, in the scope of the proposed scenario, it is assumed that C and RS may not have connectivity with AS when the access request is sent. This may be due to the use of devices with limited battery power, mobile devices taking part in the communication, or the inherent nature of the common networks which are deployed on constrained IoT environments.

The proposed scenario is split into two main phases. During the bootstrapping stage (steps 1-17), the device tries to get access to the network through an authentication process based on SEAPOL, EAP and RADIUS. Additionally, this phase includes a process by which a set of privileges are inferred and sent to the device by using capability tokens, which are

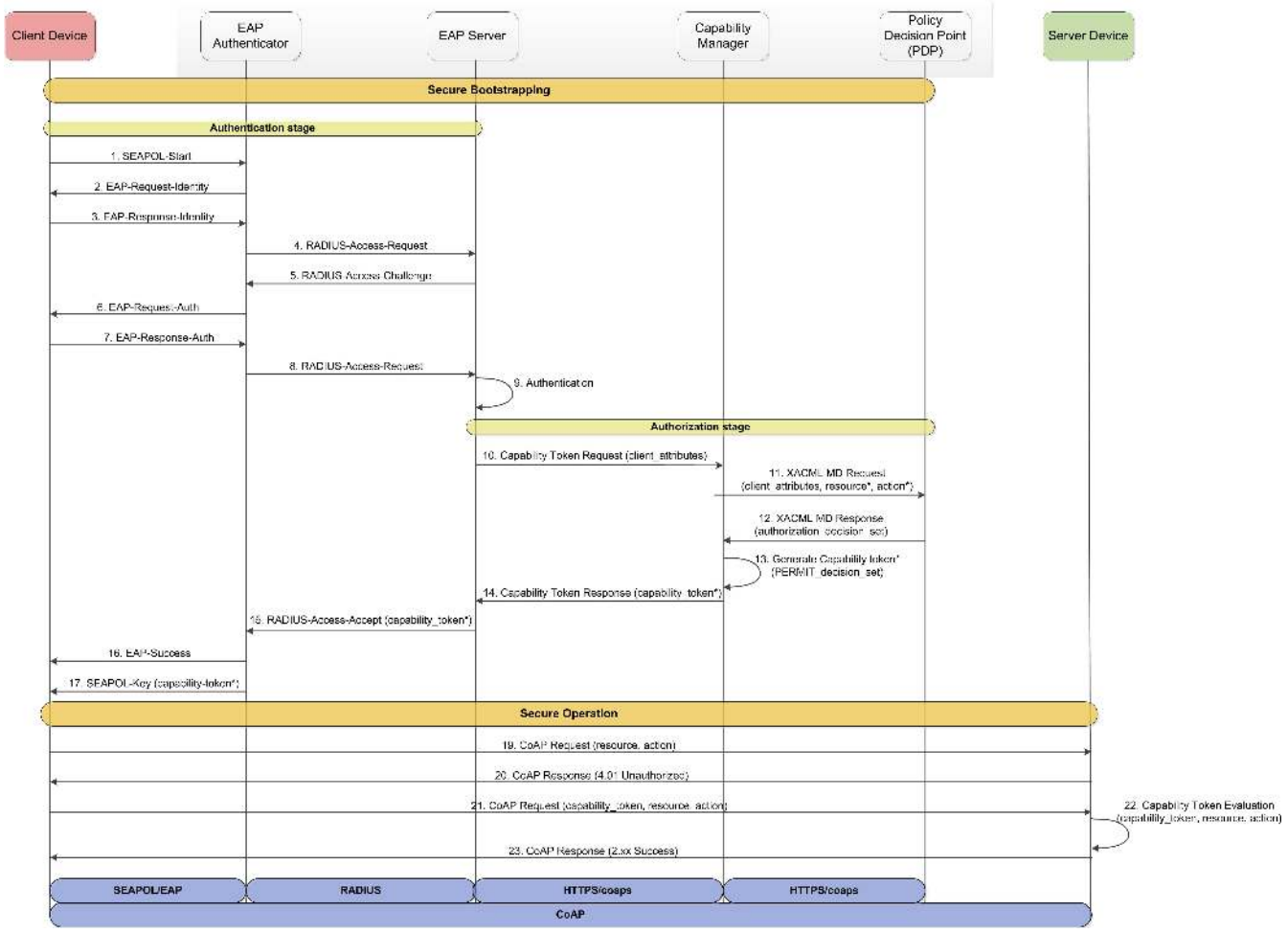

Figure 6: Bootstrapping and operational stages overview

used at the operation stage (steps 18-23). The bootstrapping process comprises, in turn, two different stages. During the *authentication phase*, the device wishing to get access to the network, acting as a *Supplicant*, sends a *SEAPOL-Start* message according to the description provided at Section IV-A. This request is answered by the EAP authenticator through an *EAP-Request-Identity* message. In order to continue the authentication process, the device acting as an EAP peer, sends an *EAP-Response-Identity* message. The content of this packet depends on the EAP method being used. In this case, an authentication method based on the use of certificates (e.g. TLS [60] or a simplified version of it), which are given to devices during the provisioning stage, is assumed. Once the EAP authenticator (operating in pass-through mode) receives this message, it encapsulates it within an *EAP-Message* attribute, which is sent to the EAP server by using a *RADIUS Access-Request* packet. Then, the EAP server issues a *RADIUS Access-Challenge* message in order to obtain the required credential from the device to be successfully authenticated. Depending on the EAP method being used, this message exchange can be repeated until the authentication process is completed. Additionally, as a result of a successful authentication, the EAP method can export two keys: the MSK and the EMSK. These keys are derived between the EAP peer (device) and the EAP server, and can be employed in a subsequently negotiated ciphersuite.

According to the proposed scenario, once the device is successfully authenticated, the bootstrapping process continues with the authorization stage. For this purpose, the RADIUS server issues a request to the CapM in order to obtain authorization credentials to be delivered to the device. This request includes the identity attributes contained in the client certificate, which is obtained during the previous stage. Examples of identity attributes for an IoT device could include: owner, device class or hardware features. This message is sent to the CapM, which is responsible for generating authorization credentials. Furthermore, this communication is assumed to be performed between non-constrained devices; consequently, it can be secured by standard mechanisms (e.g. HTTPS). In order to generate the authorization token, the CapM needs to obtain the set of privileges which are granted to the device. However, unlike a typical access control scenario in which an authorization request contains the 3-tuple (subject, resource, action), in this case only the identity attributes of the device (subject) are known. Therefore, under the core XACML foundations, this would require sending as many requests as combinations of resources and actions exist. In our scenario, this issue is tackled by employing the XACML

Multiple Decision Profile (MDP) [59], which allows the CapM (acting as a *Policy Enforcement Point* (PEP)) to formulate a set of authorization queries with a single request. The motivation behind the use of MDP is that on a common deployment, the PDP is running on a separate network component. While the communication between CapM and PDP is assumed to be performed through non-constrained networks, an environment with a huge number of resources could require a lot of authorization requests, and consequently, a higher delay and network resources. In particular, the MDP request is based on the "Repeated <Attributes> categories" type [59], which makes use of multiple <Attributes> elements with repeated category in a request context to specify a request for access to multiple decisions. Therefore, in the proposed scenario, the content of this request includes a 3-tuple consisting of: subject (client device attributes), resource (set of resources identified by <Attributes> elements in the Resource category) and action (set of actions identified by <Attributes> elements in the Action category). Resources are usually identified by a URI (e.g. "*coap://sensor1/temperature*"), while the actions considered are directly mapped to the set of CoAP methods (i.e. GET, POST, PUT, DELETE).

When the PDP receives this message, it evaluates the MDP request by generating a set of individual authorization queries. Each of these queries is defined by the elements of the cartesian product of the sets of attributes composing the resource and action categories. The set of authorization decisions which are obtained during this process are sent to the CapM in a single message. Then, it generates an authorization token with the subset of *'PERMIT'* authorization decisions which was returned by the PDP in the previous message. Additionally, the token is signed by CapM in order to avoid security breaches. The format of this credential is based on the capability token defined in [18]. Depending on the specific scenario, the set of privileges to be embedded into the token could be large, and consequently, the size of the token may be oversized to be sent on constrained environments. In that case, the set of privileges can be split up into a set of tokens, each containing a subset of access rights. Then, the CapM sends the authorization token(s) to the RADIUS server by using a *RADIUS-Access-Accept* message. This message is sent to the EAP authenticator, indicating that the device has been successfully authenticated. Finally, it sends two messages to the device. An *EAP-Success* packet to confirm the devices was successfully authenticated, and a *SEAPOL-Key* message including the authorization token, which was previously obtained.

The operational stage begins when the *client* device (C) tries to access a hosted service on another device acting as a *resource server* (RS). During this phase, constrained devices can communicate by using the authorization credentials previously obtained, in order to exchange information in a secure way. It should be pointed out that, unlike the bootstrapping process, the communication at this stage is intended to be performed between constrained devices. Consequently, the required message exchange for this phase has been designed taking into account the requirements of IoT constrained environments [51]. This process is based on CoAP, which is secured by a simplified version of DTLS based on ECDHE-ECDSA [57].

Algorithm 1 Capability Token Evaluation process

Require: Encrypted CoAP Request $ereq$ containing: Capability token of client c: ct;
Session key sk_x previously established;
Security Context sc with the association (sk_x, spk_c);

req=decrypt($sk_x, ereq$)
if association(sc, sk_x)=spk_c **then**
  if (currenttime $> ct$.nb $> ct$.ii) AND (currenttime $< ct$.na) **then**
    for all accessright $ar \in ct$.accessrights **do**
      if (ar.action=req.method=$'POST'$ AND ar.resource=req.LocationPath) OR (ar.resource=req.options.$requestUri$) **then**
        if conditions $co \in ar$ are fulfilled **then**
          permittedAction = true
          break
        end if
      end if
    end for
    if permittedAction **then**
      if verifysignature(ct.si,pk_{capm}) = true **then**
        Authorized Token
      end if
    end if
  end if
end if

As mentioned above, it is assumed that each device has a certificate with an associated key pair: (spk_c, ssk_c) for C, and (spk_s, ssk_s) for RS. In addition, RS has the public key of CapM (pk_{capm}), which is intended to manage the access to devices within the network.

Before the first message is sent, C generates an ephemeral key pair (epk_c, esk_c), in which $epk_c = esk_c * G$ and G is the curve generator supported by RS. This information can be embedded in the authorization token or obtained by C before the communication. Then, C sends a CoAP request containing both its static public key spk_c and the ephemeral public key epk_c. This message is signed by C with ssk_c using the ECDSA. In addition, timestamps are used in order to avoid replay attacks. When RS receives this message, the signature is verified by making use of spk_c, which is attached into the request. Then, RS generates an ephemeral key pair (epk_s, esk_s) where $epk_s = esk_s * G$. Thus, RS computes $(sk_x, sk_y) = esk_s * epk_c$, where sk_x is the shared secret between RS and C, which is used to send the token over a protected channel. Additionally, RS creates a security context in which the association between sk_x and spk_c is set. Then, RS sends a Coap Response (4.01 Unauthorized) indicating C is not authorized to perform the requested action. This message includes the epk_s key, and is signed with ssk_s by using ECDSA. When C receives this message, it validates the signature by employing spk_s, which was obtained before this message exchange. In the same message, C receives epk_s and, consequently, it can calculate $(sk_x, sk_y) = esk_c * epk_s$, where sk_x is the secret key, which is shared with RS. At the beginning of the second message exchange, both entities hold sk_x to be used as a session key. Therefore, the authorization token is sent over a protected channel for privacy concerns. In this way, C sends a CoAP request containing the token, which is encrypted with sk_x. When RS receives this message, it runs the authorization process which is described in the Algorithm 1 based on the description provided by [18]. After the request is decrypted with sk_x, it checks the associated public key which was previously set in the security context. Then, the authorization token is evaluated. Taking into account the constrained nature of devices, this evaluation is performed according to the complexity of the required operations. Firstly, the device checks the token is valid, that is, if it has not expired. Then, for each one of the privileges which are contained in the

token, the device checks if the requested action of the CoAP message matches the action of the access right. In addition, contextual conditions are analysed. In the case that one of these verifcations fails, the next action right element is taken for evaluation, if any. Before granting the permission, the CapM signature for this token has to be verified. For this purpose, RS makes use of the CapM's public key pk_{capm}. This phase is left to the end due to the cost required by the cryptographic operations. Finally, if the signature is successfully evaluated, the request is permitted. Consequently, RS sends a CoAP response (with 2.xx code), which is protected again with sk_x.

VI. Evaluation Results

A. Bootstrapping Stage

For the first stage of the proposed scenario, several tests have been performed in which EAPOL and SEAPOL over 802.15.4 networks have been analysed and compared. These tests were performed on Tmote Sky modules with MSP430 microcontrollers, with 10KB of RAM and 48 KB of flash memory. One mote was connected to a regular PC through a USB port acting as an Access Point, and other motes acted as clients attempting to authenticate and connect to the network. All motes were using Contiki OS 3.x.

	NullMAC	EAPOL	SEAPOL
slip-radio	71906	75552	75189
slip-radio (flash)	30618	33380	33110
sky-websense	91293	97896	96789
sky-websense (flash)	45390	49144	48808
native-border-router	413177	453157	453157

Figure 7: Memory comparison (in bytes) between EAPOL and SEAPOL for Contiki OS applications

In particular, three applications were modified: *native-boarder-router*, *slip-radio* and *sky-websense*. In order to optimize the memory footprint and consider a MAC layer without IEEE 802.15.4 design constraints, the lightweight *NullMAC* driver (without MAC layer processing) was chosen and extended to support two separate roles of Supplicant and Authenticator. Furthermore, the functionality of EAP layer was added with only MD5 as an EAP method to verify the correctness of this design. In addition, a lightweight RADIUS communication layer was implemented to support the message exchange with the FreeRADIUS server through the IPv6 protocol. These comparisons were done taking into account the NullMAC and NullRDC drivers. According to the results in Figure 7, modifications to the native-border-router increased its size around 9.68% in relation to the NullMAC driver. This change is mostly due to the fact of the implementation of RADIUS communication mechanisms. The introduced modifications are the same for the EAPOL and SEAPOL solutions because the native-border-router does not use directly the MAC driver; it communicates with the 802.15.4 network through the access point through the slip-radio application. The slip-radio modification of the EAPOL MAC driver increased the size around 9% of the flash memory usage in relation to the NullMAC/RDC drivers. The optimized version of the EAPOL protocol (SEAPOL) reduced the usage of flash memory by 270 bytes. Moreover, the set of modifications of the sky-websense application increased the flash memory usage by 3754 bytes for EAPOL MAC (that is, around a 12.2% increase of memory usage). The SEAPOL MAC version decreased the flash memory usage by 336 bytes (3418 used bytes), which represents a 11.1% decrease in comparison to the EAPOL MAC driver. It should be pointed out that during all of these tests, Supplicant devices were able to successfully authenticate with regular FreeRADIUS server by using EAP-MD5 as an authentication mechanism.

Moreover, we analysed the impact of SEAPOL to different EAP methods in relation to the standard EAPOL according to implementation size and network usage. During this process, EAP-MD5, EAP-PSK and EAP-TLS were successfully implemented and tested on the testbed previously described. Regarding the memory usage, the results are shown in Figure 8, in which different EAP methods are considered. According to the results, EAP-MD5 is the most lightweight EAP Method. It requires only 2498 bytes of ROM for SEAPOL (11.98% memory usage reduction in comparison to the regular EAPOL version) and 65 bytes of RAM. EAP-PSK [33] requires 9614 bytes of ROM (3.41% memory usage reduction in comparison to the regular EAPOL version) and 226 bytes of RAM. It provides a higher security level than MD5 but is only applicable to scenarios with a small number of devices. The second category of EAP methods which were analysed consists of EAP-TLS methods with ECDSA as an authentication scheme. Different configurations of these methods require a relatively moderated amount of RAM memory (5169 bytes for the SECG-P160 curve, and 5700 bytes for the NIST-P256 curve). These methods represent the category with the highest ROM memory requirements. Specifically, they require 29 and 33 kilobytes for TLS-1.0 (with a reduction of 1.09% and 1.01% of ROM through SEAPOL) and 35 and 39 kilobytes for the TLS-1.2 version (with 0.96% and 0.86% reduction in the case of SEAPOL), respectively. Last category of the analysed EAP methods consists of EAP-TLS methods with RSA as an authentication scheme. In this case, while they require less ROM memory than ECDSA version (with a 1.50% and 1.18% reduction with SEAPOL in relation to EAPOL), RAM memory requirements are significantly higher. Consequently, while they provide a similar security regarding ECDSA-based methods, they are hardly applicable to devices with tight memory constraints.

ContikiOS MAC driver	EAPOL		SEAPOL		Memory usage reduction	
	ROM	RAM	ROM	RAM	ROM	RAM
nullmac	17820	1410	17820	1410	0%	0%
EAP-NULL	+832	+64	+492	+64	40.86%	0%
EAP-MD5	+2838	+65	+2498	+65	11.98%	0%
EAP-PSK	+9954	+226	+9614	+226	3.41%	0%
EAP-TLS-1.0-ECDSA-160	+29335	+5169	+29015	+5169	1.09%	0%
EAP-TLS-1.2-ECDSA-160	+35575	+5169	+35235	+5169	0.96%	0%
EAP-TLS-1.0-ECDSA-256	+33758	+5700	+33418	+5700	1.01%	0%
EAP-TLS-1.2-ECDSA-256	+39978	+5700	+39638	+5700	0.86%	0%
EAP-TLS-1.0-RSA-480	+22546	+9278	+22206	+9278	1.50%	0%
EAP-TLS-1.2-RSA-480	+28766	+9278	+28426	+9278	1.18%	0%
EAP-TLS-1.0-RSA-512	+22546	+9509	+22206	+9509	1.50%	0%
EAP-TLS-1.2-RSA-512	+28766	+9509	+28426	+9509	1.18%	0%
EAP-TLS-1.0-RSA-1024	+22546	+13153	+22206	+13153	1.50%	0%
EAP-TLS-1.2-RSA-1024	+28766	+13153	+28426	+13153	1.18%	0%
EAP-TLS-1.0-RSA-2048	+22546	+20567	+22206	+20567	1.50%	0%
EAP-TLS-1.2-RSA-2048	+28766	+20567	+28426	+20567	1.18%	0%

Figure 8: Memory usage comparison between EAPOL and SEAPOL

Additionally, we studied the network usage different EAP methods according to EAPOL and SEAPOL implementations. The set of obtained results is shown in Figure 9. Firstly, the EAP-MD5 over EAPOL requires the transmission of 6 packets with 161 bytes which is reduced by SEAPOL, requiring 125 bytes (a 22.36% reduction). The EAP-PSK used 9 packets with 341 bytes which represents a 13.67% reduction of transmitted data in comparison to the EAPOL standard. Both ECDSA configurations of TLS require 1083 and 1217 bytes of network traffic with around 30 packets. SEAPOL provides a 13.84% and 13.26% network traffic reduction for SECG-P160 and NIST-P256 curves, respectively. Moreover, RSA-based configurations of TLS require 19, 20, 27 and 43 packets, respectively. By using SEAPOL, 12%, 11.76%, 10.99% and 10% network traffic reduction was obtained.

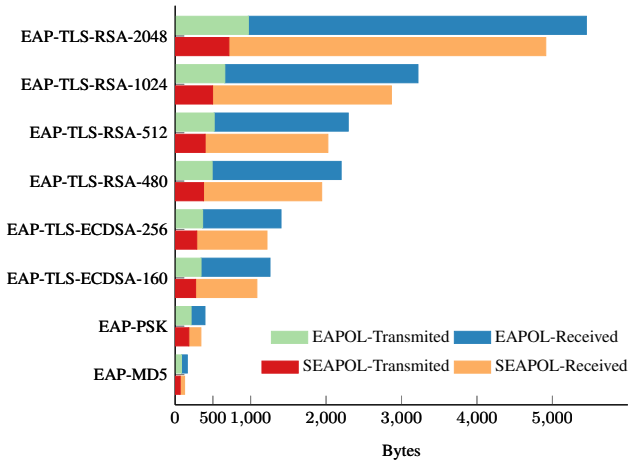

Figure 9: Network usage comparison between EAPOL and SEAPOL

According to the results, ECDSA-based TLS methods provide a more suitable trade-off between security properties and IoT constraints. These methods were implemented over SEAPOL, providing an lightweight version of EAPOL in terms of memory and network usage, which are required in order to deploy security mechanisms on constrained IoT environments. Additionally, SEAPOL can be leveraged for the design and development of IoT network architectures in order to deploy lightweight bootstrapping mechanisms, as the proposed scenario in this work.

B. Operational Stage

For the second stage of the proposed scenario, the required message exchange and the token evaluation have been implemented. Specifically, C and RS have been developed over JN5148 motes equipped with Contiki OS, programmable CPU, and a unified memory architecture which contains 128 Kbytes of ROM, 128 Kbytes of RAM and a 32-byte One Time Programmable (OTP) eFuse memory.

At the operation stage, the memory usage which was required for the designed functionality is shown in Figure 10. It should be noted that these values makes reference to the device acting as a resource server (RS) in the proposed

Module	ROM	ROM increase	RAM	RAM increase
Contiki 2.6	51002	0%	11644	0%
CoAP server	53030	3.98%	12036	3.37%
JSON parser	57177	7.82%	12156	1%
Capability Token processing	57531	0.62%	12156	0%
MD5	60231	4.7%	12156	0%
base64 scheme	6164	1.55%	121556	0%
ECC library	70202	14.78%	12336	1.48%
Total	70202	37.65%	12336	5.94%

Figure 10: Code size for software modules for operation security

scenario. These results were obtained according to the gradual addition of the different modules, which are required for our application. For this purpose, we used the ba-elf-size tool that is provided by the ba-elf2 compiler for the "ba2" instructions set of the JN5148 OpenRISC architecture According to the results, the implementation of the components represents a 37.65% increase of ROM and a 5.94% of RAM, in relation to an empty application for Contiki OS. As expected, the component that needs more resources is the ECC library, representing a 14.78% increase of ROM. Other heavy components are related to the CoAP server functionality, as well as JSON parser, which is required to parse the authorization token. Nevertheless, our lightweight design and ECC optimizations makes it possible to embed authentication and authorization functionality on devices with tight resource constraints.

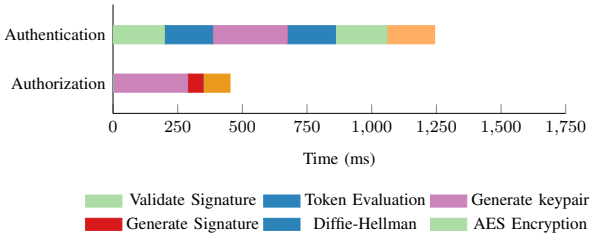

Figure 11: Time results for operation security

Moreover, in order to demonstrate the feasibility of mechanism, 100 tests were executed of the proposed message exchange. The mean values of the execution times are shown in Figure 11. As discussed in Section V, in the first message exchange C and RS are mutually authenticated and, as a result of this step, a session key is established to send the authorization token. Therefore, this stage includes the following tasks:

- Sign message (*198.23 ms*): it includes the delay required to sign a CoAP message by using ECDSA.
- Validate signature (*288.18 ms*): time required to verify the ECDSA signature from C or RS.
- Generate keypair (*186.74 ms*): it includes the time to generate an ephemeral keypair.
- Generate session key (*184.8 ms*): includes the delay required to generate the session key from ephemeral a public key and ephemeral secret key.

These operations are performed by C and RS during the first message exchange. According to results, the mean time required for the mutual authentication stage is 1715.9 ms $(857.95ms * 2)$. At this point, it should be pointed out that some of these operations could be performed in parallel (e.g. session key generation by RS) while messages are over the

link. Furthermore, while this time includes the set of required operations by C and RS, the Round-Trip-Time (RTT) for this exchange would not include some of these tasks (e.g. the initial signature by C). For the second message exchange, C sends a CoAP request in which the capability token is attached. Therefore, it requires the delays for:

- Symmetric encryption/decryption (*104.38 ms*): the delay which is required for symmetric cryptography operations by using *Advanced Encryption Standard* (AES) [61]. It includes the time to encrypt and decrypt both messages.
- Token evaluation (59.7 ms): includes the delay to check if the user is authorized to perform the requested action and if the conditions are met.
- Validate signature (*288.18 ms*): time required to verify the ECDSA signature which is contained in the capability token (CapM's signature).

Consequently, the mean time which is required for the second stage is 452.26 ms. Therefore, taking into account the results of the previous phase, the delay of the whole process for secure operation is 2168.16 ms. At this point, it should be noted that this time assumes that C and RS are successfully authenticated, and C is authorized to perform the requested action and, therefore, all steps of the mechanism must be completed. As expected, the most expensive tasks are related to public-key cryptography operations. However, unlike most of previous security proposals for IoT which are based on symmetric cryptography, our optimized ECC library has enabled to embed this functionality into constrained devices, improving results over more consolidated ECC implementations, such as TinyECC [26] and NanoECC [62]. In addition, our lightweight design has allowed the development of a decentralized authentication and authorization mechanism, providing the benefits of a distributed approach in terms of flexibility, interoperability and end-to-end security.

VII. Discussion

The seamless integration of physical objects in the Internet infrastructure requires the application of lightweight security mechanisms to be used even in constrained environments. Current standard security solutions were not designed taking into account the inherent nature of such IP-based global ecosystem, and consequently, a deep revision and adaptation of these mechanisms need to be considered. These challenges have attracted more and more attention from research community and recently several efforts are starting to emerge in this direction. In particular, the IETF *Authentication and Authorization for Constrained Environments* (ACE) WG was recently established to accommodate the current standard security mechanisms to IoT environments with tight resource constraints.

On the one hand, ACE WG is focused on the definition of requirements and considerations that must be addressed by security mechanisms that are designed for constrained environments. Specifically, the work presented in [51] provides a set of assumptions and requirements which were analysed and mostly addressed by the integral scenario proposed in this work. On the other hand, [58] provides an initial architecture which is composed by a set of actors with different requirements and functionality. In particular, the scenario considered by ACE is based on a *client* (C), which wants to get access to a *resource* (R) hosted in a *resource server* (RS). As in the proposed scenario, C and RS are considered as constrained level actors, which communicate over a constrained level protocol (i.e. CoAP). Moreover, the architecture defines an *Authentication Manager* (AM) and an *Authorization Server* (AS), which are supposed to be less-constrained level actors. In particular, the AS functionality is performed by the Capability Manager and the PDP in the proposed scenario; furthermore, the functionality of the AM is carried out by the AAA infrastructure. These entities are integrated in order to generate proper credentials for authenticated and authorized entities, which can be employed by constrained devices for secure operation. Also, our lightweight authentication and authorization mechanisms provide a flexible approach, allowing the specification of temporal privileges and delegation, in order to fit with the use cases which are considered in [63].

In addition to the main considerations of ACE WG, our scenario includes a security bootstrapping process in order to obtain authorization credentials to be used during the operational stage. This mechanism is based on standard technologies and was optimized to be used over constrained environments. Specifically, it is based on a lightweight version of EAPOL (SEAPOL) to transport EAP messages, achieving a proper security mechanism which does not require additional protocols or message exchange. While it was proven as a suitable approach for constrained devices, it requires obtaining a credential or set of credentials containing all the privileges for a device at the bootstrapping stage. On the one hand, with this solution, a device does not have to request new authorization credentials during their life cycle, which is a valuable feature for constrained devices and Low power and Lossy Networks (LLNs). On the other hand, the set of privileges for a specific device can change, and the size of credentials could be too large to be sent over LLNs, if this issue is not properly addressed. Currently, other protocols could be employed as an alternative to get security credentials to be employed by constrained devices on IoT scenarios. Specifically, PANA [29] uses notification messages (PNR/PNA) for signaling reauthentication and performing liveness test, which are sent at any time during the access phase. The semantics of these messages could be extended by defining new Attribute-Value Pairs (AVPs), which could be used by PANA Clients (PaC) to request authorization credentials. In this case, while a device could get credentials according to their needs at any time, obtaining an authorization token for a specific transaction could hinder revocation tasks, as well the credentials management for constrained devices.

VIII. Conclusions and Future Work

The realization of IoT scenarios requires addressing significant security implications, since everyday devices are being integrated into the Internet infrastructure. While in recent years, technological challenges were overcome through the extension and adaptation of wireless communication technologies, IoT paradigm has to cope with challenges related

to the application of security mechanisms over constrained environments. Under these considerations, this work proposes a set of lightweight authentication and authorization mechanisms in order to embed authentication and authorization functionality on constrained smart objects. These mechanisms are integrated and extended with other standard technologies in order to address different security planes of the life cycle of an IoT device within an ARM-compliant security framework to be deployed on IoT scenarios. In addition, we present a set of evaluation results in order to demonstrate the feasibility of the proposed scenario, by optimizing different aspects of these mechanisms. Future work is focused on the integration of the proposed solution into the IETF ACE WG as well as the definition of alternative scenarios, in order to assess and compare the suitability of such scenario. Specifically, the design and development of standard-based alternative mechanisms, such as PANA, will be analysed in order to obtain a trade-off between the features which are provided by different solutions. Moreover, we will analyse the design and development of other lightweight security solutions addressing other components of the proposed framework, in order to provide a comprehensive security approach for constrained IoT environments.

Acknowledgments

This work has been sponsored by European Commission through the FP7-SMARTIE-609062 and the FP7-SOCIOTAL-609112 EU Projects, the Spanish Seneca Foundation by means of the Excellence Researching Group Program (04552/GERM/06) and the FPI program (grant 15493/FPI/10). In addition, it has been supported by the Swiss national government through the Sciex-NMSch (Scientific Exchange Programme between Switzerland and the New Member States of the EU) with the project code 13.121, named BASTION "Bootstrapping, Authentication, Security and Trust for the Internet of Things Networks".

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," *Elsevier Computer Networks*, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] V. Mayer-Schönberger and K. Cukier, *Big data: A revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt, 2013.

[3] H. Schaffers, N. Komninos, M. Pallot, B. Trousse, M. Nilsson, and A. Oliveira, *Smart cities and the future internet: Towards cooperation frameworks for open innovation*. Springer, 2011.

[4] N. Kushalnagar, G. Montenegro, and C. Schumacher, "Ipv6 over low-power wireless personal area networks (6lowpans): overview, assumptions, problem statement, and goals," *RFC4919, August*, vol. 10, 2007.

[5] A. J. Jara, M. A. Zamora, and A. Skarmeta, "Glowbal ip: An adaptive and transparent ipv6 integration in the internet of things," *Mobile Information Systems*, vol. 8, no. 3, pp. 177–197, 2012.

[6] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," *IETF RFC 7252*, vol. 10, June 2014.

[7] A. J. Jara, A. C. Olivieri, Y. Bocchi, M. Jung, W. Kastner, and A. F. Skarmeta, "Semantic web of things: an analysis of the application semantics for the iot moving towards the iot convergence," *International Journal of Web and Grid Services*, vol. 10, no. 2, pp. 244–272, 2014.

[8] C. P. Mayer, "Security and privacy challenges in the internet of things," *Electronic Communications of the EASST*, vol. 17, 2009.

[9] C. M. Medaglia and A. Serbanati, "An overview of privacy and security issues in the internet of things," in *The Internet of Things*. Springer, 2010, pp. 389–395.

[10] D. Miorandi, S. Sicari, F. Pellegrini, and I. Chlamtac, "Internet of Things: Vision, Applications & Research Challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497–1516, September 2012.

[11] R. Roman, J. Zhou, and J. Lopez, "On the features and challenges of security and privacy in distributed internet of things," *Computer Networks*, vol. 57, no. 10, pp. 2266–2279, 2013.

[12] T. Heer, O. Garcia-Morchon, R. Hummen, S. L. Keoh, S. S. Kumar, and K. Wehrle, "Security challenges in the ip-based internet of things," *Wireless Personal Communications*, vol. 61, no. 3, pp. 527–542, 2011.

[13] H. Yu, J. He, T. Zhang, P. Xiao, and Y. Zhang, "Enabling end-to-end secure communication between wireless sensor networks and the internet," *World Wide Web*, pp. 1–26, 2013.

[14] L. S. Committee *et al.*, "Part 11: wireless lan medium access control (mac) and physical layer (phy) specifications. ieee std 802.11 i-2004," *IEEE Comput Soc*, 2004.

[15] B. Aboba, L. Blunk, J. Vollbrecht, J. Carlson, H. Levkowetz *et al.*, "Extensible authentication protocol (eap)," RFC 3748, June, Tech. Rep., 2004.

[16] C. Rigney, S. Willens, A. Rubens, and W. Simpson, "Rfc 2865: Remote authentication dial in user service," *Internet Society (Jun. 2000)*, 2000.

[17] E. Rissanen, "extensible access control markup language (xacml) version 3.0 oasis standard," 2012.

[18] J. L. Hernández-Ramos, A. J. Jara, L. Marín, and A. F. Skarmeta, "Dcapbac: Embedding authorization logic into smart things through ecc optimizations," *International Journal of Computer Mathematics*, no. just-accepted, pp. 1–22, 2014.

[19] A. Bassi, M. Bauer, M. Fiedler, T. Kramp, R. van Kranenburg, S. Lange, and S. Meissner, "Enabling things to talk," 2013.

[20] S. Babar, P. Mahalle, A. Stango, N. Prasad, and R. Prasad, "Proposed security model and threat taxonomy for the internet of things (iot)," in *Recent Trends in Network Security and Applications*. Springer, 2010, pp. 420–429.

[21] A. F. Skarmeta, J. L. Hernandez-Ramos, and M. Moreno, "A decentralized approach for security and privacy challenges in the internet of things," in *Internet of Things (WF-IoT), 2014 IEEE World Forum on*. IEEE, 2014, pp. 67–72.

[22] R. Roman, P. Najera, and J. Lopez, "Securing the internet of things," *Computer*, vol. 44, no. 9, pp. 51–58, 2011.

[23] R. H. Weber, "Internet of things–new security and privacy challenges," *Computer Law & Security Review*, vol. 26, no. 1, pp. 23–30, 2010.

[24] P. Porambage, C. Schmitt, P. Kumar, A. Gurtov, and M. Ylianttila, "Pauthkey: A pervasive authentication protocol and key establishment scheme for wireless sensor networks in distributed iot applications," vol. 14, 2014.

[25] E. Rescola and N. Modadugu, "Rfc 4347: Datagram transport layer security (dtls)," *Request for Comments, IETF*, 2006.

[26] A. Liu and P. Ning, "Tinyecc: A configurable library for elliptic curve cryptography in wireless sensor networks," in *Information Processing in Sensor Networks, 2008. IPSN'08. International Conference on*. IEEE, 2008, pp. 245–256.

[27] B. Sarikaya, Y. Ohba, R. Moskowitz, Z. Cao, and R. Cragie, "Security bootstrapping solution for resource-constrained devices," *IETF Internet Draft, draft-sarikaya-core-sbootstrapping-05*, 2012.

[28] R. Hummen, J. Hiller, M. Henze, and K. Wehrle, "Slimfit—a hip dex compression layer for the ip-based internet of things," in *Wireless and Mobile Computing, Networking and Communications (WiMob), 2013 IEEE 9th International Conference on*. IEEE, 2013, pp. 259–266.

[29] D. Forsberg, Y. Ohba, B. Patil, H. Tschofenig, and A. Yegin, "Rfc 5191 protocol for carrying authentication for network access (pana)," *Network Working Group*, 2008.

[30] "8802-1x-2013," 2013. [Online]. Available: http://dx.doi.org/10.1109/ieeestd.2013.6679204

[31] O. Garcia-Morchon, S. Keoh, S. Kumar, P. Moreno-Sánchez, F. Vidal-Meca, and J. Ziegeldorf, "Securing the IP-based internet of things with HIP and DTLS," in *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*. ACM, 2013, pp. 119–124.

[32] P. M. Sanchez, R. M. Lopez, and A. F. G. Skarmeta, "Panatiki: A network access control implementation based on pana for iot devices," *Sensors*, vol. 13, no. 11, pp. 14 888–14 917, 2013.

[33] F. Bersani and H. Tschofenig, "Rfc 4764-the eap-psk protocol: A pre-shared key extensible authentication protocol (eap) method," IETF Network Working Group, Tech. Rep., 2007.

[34] J. Liu, Y. Xiao, and C. L. P. Chen, "Authentication and Access Control in the Internet of Things," in *Proc. of the 32nd International Conference on Distributed Computing Systems Workshops (ICDCSW), Macau, China*. IEEE, June 2012, pp. 588–592.

[35] D. Ferraiolo, J. Cugini, and R. Kuhn, "Role-based access control (RBAC): Features and motivations," in *Proc. of 11th Annual Computer Security Application Conference*, 1995, pp. 241–48.

[36] B. Ndibanje, H.-J. Lee, and S.-G. Lee, "Security analysis and improvements of authentication and access control in the internet of things," *Sensors*, vol. 14, no. 8, pp. 14 786–14 805, 2014.

[37] N. YE, Y. Zhu, R. chuan WANG, R. Malekian, and L. Qiao-min, "An efficient authentication and access control scheme for perception layer of internet of things," *Applied Mathematics & Information Sciences*, vol. 8, no. 4, pp. 1617–1624, jul 2014. [Online]. Available: http://dx.doi.org/10.12785/amis/080416

[38] E. Yuan and J. Tong, "Attributed based access control (ABAC) for web services," in *Proc. of the 12th IEEE International Conference on Web Services (ICWS), Orlando, USA*. IEEE, July 2005.

[39] P. N. Mahalle, B. Anggorojati, N. R. Prasad, and R. Prasad, "Identity Establishment and Capability based Access Control (iecac) scheme for Internet of Things," in *Proc. of the 15th International Symposium on Wireless Personal Multimedia Communications (WPMC), Taipei, China*. IEEE, September 2012, pp. 187–191.

[40] S. Gusmeroli, S. Piccione, and D. Rotondi, "A capability-based security approach to manage access control in the internet of things," *Mathematical and Computer Modelling*, vol. 58, no. 5-6, pp. 1189–1205, September 2013.

[41] L. Seitz, G. Selander, and C. Gehrmann, "Authorization framework for the internet-of-things," in *2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. IEEE, jun 2013. [Online]. Available: http://dx.doi.org/10.1109/wowmom.2013.6583465

[42] D. Crockford, "RFC 4627: The application/json Media Type for Javascript Object Notation (JSON)," IETF RFC 4627, July 2006, http://www.ietf.org/rfc/rfc4627.txt.

[43] L. Marin, A. J. Jara, and A. F. Skarmeta, "Shifting primes: Optimizing elliptic curve cryptography for 16-bit devices without hardware multiplier," *Mathematical and Computer Modelling*, 2013.

[44] A. J. Jara, "Trust extension protocol for authentication in networks oriented to management (tepanom)," in *Availability, Reliability, and Security in Information Systems*. Springer, 2014, pp. 155–165.

[45] M. Bauer, E. Berg, F. Carrez, S. Ebers, S. Haller, T. Jacobs, S. Krco, F. Massel, G. Woysch, and R. Egan, "The internet of things initiative (iot-i). deliverable 1.5: Iot reference model white paper," 2011.

[46] S. Ziegler, C. Crettaz, L. Ladid, S. Krco, B. Pokric, A. F. Skarmeta, A. Jara, W. Kastner, and M. Jung, *Iot6–moving to an ipv6-based future iot*. Springer, 2013.

[47] "Butler. deliverable 3.2: Integrated system architecture and initial pervasive butler proof of concept," 2013.

[48] V. Tsiatsis, A. Gluhak, T. Bauge, F. Montagut, J. Bernat, M. Bauer, C. Villalonga, P. M. Barnaghi, and S. Krco, "The sensei real world internet architecture." in *Future Internet Assembly*, 2010, pp. 247–256.

[49] S. Sotiriadis, E. G. Petrakis, S. Covaci, P. Zampognaro, E. Georga, and C. Thuemmler, "An architecture for designing future internet (fi) applications in sensitive domains: Expressing the software to data paradigm by utilizing hybrid cloud technology," in *Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on*. IEEE, 2013, pp. 1–6.

[50] J. B. Bernabe, J. L. Hernández, M. V. Moreno, and A. F. S. Gomez, "Privacy-preserving security framework for a social-aware internet of things," in *Ubiquitous Computing and Ambient Intelligence. Personalisation and User Adapted Services*. Springer, 2014, pp. 408–415.

[51] L. Seitz and G. Selander, "Problem description for authorization in constrained environments," *IETF Internet Draft, draft-seitz-ace-problem-description-01*, 2014.

[52] C. Kaufman, P. Hoffman, Y. Nir, and P. Eronen, "Internet key exchange protocol version 2 (ikev2)," RFC 5996, September, Tech. Rep., 2010.

[53] A. J. Jara, D. Fernandez, P. Lopez, M. A. Zamora, and A. F. Skarmeta, "Lightweight mipv6 with ipsec support," *Mobile Information Systems*, vol. 10, no. 1, pp. 37–77, 2014.

[54] S. Raza, T. Voigt, and V. Jutvik, "Lightweight ikev2: A key management solution for both the compressed ipsec and the ieee 802.15. 4 security," in *Proceedings of the IETF Workshop on Smart Object Security*, 2012.

[55] J. Dennis and E. V. Horn, "Programming Semantics for Multiprogrammed Computations," *Communications of the ACM*, vol. 9, no. 3, pp. 143–155, 1966.

[56] L. Marin, A. J. Jara, and A. F. Gómez-Skarmeta, "Multiplication and squaring with shifting primes on openrisc processors with hardware multiplier." *J. UCS*, vol. 19, no. 16, pp. 2368–2384, 2013.

[57] S. Blake-Wilson, N. Bolyard, V. Gupta, C. Hawk, and B. Möller, "Elliptic curve cryptography (ecc) cipher suites for transport layer security (tls). internet engineering task force," *Network Working Group, RFC*, vol. 4492, 2006.

[58] S. Gerdes, "Actors in the ace architecture," *IETF Internet Draft, draft-gerdes-ace-actors-01*, 2014.

[59] E. Rissanen, "Xacml v3.0 multiple decision profile version 1.0," OASIS Committee Specification 02, Tech. Rep., 2014.

[60] D. Simon, B. Aboba, and R. Hurst, "Rfc 5216 the eap-tls authentication protocol," 2008.

[61] J. Daemen and V. Rijmen, *The design of Rijndael: AES-the advanced encryption standard*. Springer, 2002.

[62] P. Szczechowiak, L. B. Oliveira, M. Scott, M. Collier, and R. Dahab, "Nanoecc: Testing the limits of elliptic curve cryptography in sensor networks," in *Wireless sensor networks*. Springer, 2008, pp. 305–320.

[63] L. Seitz, S. Gerdes, G. Selander, M. Mani, and S.Kumar, "Ace use cases," *IETF Internet Draft, draft-seitz-ace-usecases-01*, 2014.

José L. Hernández-Ramos received the B.S. degree in 2011, and the M.S. degree in 2012 from the University of Murcia. Since 2013 he is a predoctoral researcher in the Department of Information and Communications Engineering at the same university, where he is pursuing a Ph. D degree and participating in several European research projects, such as SocIoTal and SMARTIE. His main research interests are related to the design and implementation of novel security and privacy-preserving mechanisms for the Internet of Things.

Marcin Piotr Pawlowski received M. Sc. degree in computer science from Jagiellonian University in Krakow, Poland, and B. Sc. degree in electronics and telecommunication from AGH University of Science and Technology in Krakow, Poland, in 2010 and 2013, respectively. He is pursuing Ph. D. degree in computer science at Jagiellonian University in Krakow, Poland. Recently, he has received SCIEX Fellowship for realization of the project BASTION - Analysis, design and evaluation of Bootstrapping, Authentication, Security and Trust for the Internet of Things Networks, and is working at the Institute of Information System of the University of Applied Sciences-Western Switzerland. He is interested in the network protocols, and security mechanisms, and in particularly for wireless applications such as Internet of Things.

Antonio J. Jara is Assistant Prof. PostDoc at University of Applied Sciences Western Switzerland (HES-SO) from Switzerland , vice-chair of the IEEE Communications Society Internet of Things Technical Committee, and founder of the Wearable Computing and Personal Area Networks company HOP Ubiquitous S.L., He did his PhD (Cum Laude) at the Intelligent Systems and Telematics Research Group of the University of Murcia (UMU) from Spain. He received two M.S. (Hons. - valedictorian) degrees. Since 2007, he has been working on several projects related to IPv6, WSNs. and RFID applications in building automation and healthcare. He is especially focused on the design and development of new protocols for security and mobility for Future Internet of things, which was the topic of his Ph.D. Nowadays, he continues working on IPv6 technologies for the Internet of Things in projects such as IoT6, and also Big Data and Knowledge Engineering for Smart Cities and eHealth. He has also carried out a Master in Business Administration (MBA). He has published over 100 international papers, as well, he holds one patent. Finally, he participates in several Projects about the IPv6, Internet of Things, Smart Cities, and mobile healthcare.

Antonio F. Skarmeta received the M.S. degree in Computer Science from the University of Granada and B.S. (Hons.) and the Ph.D. degrees in Computer Science from the University of Murcia Spain. Since 2009 he is Full Professor at the same department and University. He has worked on different research projects in the national and international area, like IoT6, Openlab, SWIFT, GEN6. He is associate editor of the IEEE SMC-Part B and reviewer of several international journals.

Latif Ladid holds the following positions: President, IPv6 FORUMwww.ipv6forum.org, Chair, European IPv6 Task Forcewww.eu.ipv6tf.org, Emeritus Trustee, Internet Society www.isoc.org, Board Member IPv6 Ready & Enabled Logos Program and Board Member World Summit Award www.wsis-award.org. He is a Senior Researcher at the University of Luxembourg "Security & Trust" (SnT) www.securityandtrust.lu on multiple European Commission Next Generation Technologies IST Projects. He is also a Member of 3GPP PCG (www.3gpp.org), 3GPP2 PCG (www.3gpp2.org), Vice Chair, IEEE ComSoc TCIIN , Member of UN Strategy Council, member of IEC Executive Committee and member of the Future Internet Forum EU Member States, representing Luxembourg.