Network Working Group
Internet-Draft
Intended status: Informational
Expires: June 11, 2018

O. Garcia-Morchon
Philips IP&S
S. Kumar
Philips Research
M. Sethi
Ericsson

December 08, 2017

State-of-the-Art and Challenges for the Internet of Things Security
draft-irtf-t2trg-iot-seccons-09

Abstract

The Internet of Things (IoT) concept refers to the usage of standard Internet protocols to allow for human-to-thing and thing-to-thing communication. The security needs for the IoT are well-recognized and many standardization steps for providing security have been taken, for example, the specification of Constrained Application Protocol (CoAP) over Datagram Transport Layer Security (DTLS). However, security challenges still exist and there are some use cases that lack a suitable solution. In this document, we first discuss the various stages in the lifecycle of a thing. Next, we document the various security threats to a thing and the challenges that one might face to protect against these threats. Lastly, we discuss the next steps needed to facilitate the deployment of secure IoT systems. This document can be used by IoT standards specifications as a reference for details about security considerations applying to the specified protocol.

This document is a product of the IRTF Thing-to-Thing Research Group (T2TRG).

Status of This Memo

This Internet-Draft will expire on June 11, 2018.

Copyright Notice

Table of Contents

1. Introduction

The Internet of Things (IoT) denotes the interconnection of highly heterogeneous networked entities and networks that follow a number of different communication patterns such as: human-to-human (H2H), human-to-thing (H2T), thing-to-thing (T2T), or thing-to-things (T2Ts). The term IoT was first coined by the Auto-ID center [AUTO-ID] in 1999 which had envisioned a world where every physical object is tagged with a radio-frequency identification (RFID) tag having a globally unique identifier. This would not only allow tracking of objects in real-time but also allow querying of data about them over the Internet. However, since then, the meaning of the Internet of Things has expanded and now encompasses a wide variety of technologies, objects and protocols. It is not surprising that the IoT has received significant attention from the research community to (re)design, apply, and use standard Internet technology and protocols for the IoT.

The things that are part of the Internet of Things are no longer unresponsive and have transformed into computing devices that understand and react to the environment they reside in. These things are also often referred to as smart objects or smart devices. The introduction of IPv6 [RFC6568] and CoAP [RFC7252] as fundamental building blocks for IoT applications allows connecting IoT hosts to the Internet. This brings several advantages including: (i) a homogeneous protocol ecosystem that allows simple integration with other Internet hosts; (ii) simplified development for devices that significantly vary in their capabilities; (iii) a unified interface for applications, removing the need for application-level proxies. These building blocks greatly simplify the deployment of the envisioned scenarios which range from building automation to production environments and personal area networks.

This document presents an overview of important security aspects for the Internet of Things. We begin by discussing the lifecycle of a thing and giving general definitions of the security building blocks in Section 2. In Section 3, we discuss security threats for the IoT and methodologies for managing these threats when designing a secure system. Section 4 reviews existing IP-based (security) protocols for the IoT and briefly summarizes existing guidelines and regulations. Section 5 identifies remaining challenges for a secure IoT and discusses potential solutions. Section 6 includes final remarks and

conclusions. This document can be used by IoT standards specifications as a reference for details about security considerations applying to the specified system or protocol.

The first draft version of this document was submitted in March 2011. Initial draft versions of this document were presented and discussed during the CORE meetings at IETF 80 and later. Discussions on security lifecycle at IETF 92 (March 2015) evolved into more general security considerations. Thus, the draft was selected to address the T2TRG work item on the security considerations and challenges for the Internet of Things. Further updates of the draft were presented and discussed during the T2TRG meetings at IETF 96 (July 2016) and IETF 97 (November 2016) and at the joint interim in Amsterdam (March 2017). This document has been reviewed by, commented on, and discussed extensively for a period of nearly six years by a vast majority of T2TRG and related group members; the number of which certainly exceeds 100 individuals. It is the consensus of T2TRG that the security considerations described in this document should be published in the IRTF Stream of the RFC series. This document does not constitute a standard.

2. Motivation and background

This section begins by describing the lifecycle of a thing. It then details the five different security building blocks that can be used for analyzing and classifying the security aspects of the IoT.

2.1. The Thing Lifecycle

The lifecycle of a thing refers to the operational phases of a thing in the context of a given application or use case. Figure 1 shows the generic phases of the lifecycle of a thing. This generic lifecycle is applicable to very different IoT applications and scenarios. For instance, [RFC7744] provides an overview of relevant IoT use cases.

In this document, we consider a Building Automation and Control (BAC) system to illustrate the lifecycle and the meaning of these different phases. A BAC system consists of a network of interconnected nodes that performs various functions in the domains of HVAC (Heating, Ventilating, and Air Conditioning), lighting, safety, etc. The nodes vary in functionality and a large majority of them represent resource-constrained devices such as sensors and luminaries. Some devices may be battery operated or may rely on energy harvesting. This requires us to also consider devices that sleep during their operation to save energy. In our BAC scenario, the life of a thing starts when it is manufactured. Due to the different application areas (i.e., HVAC, lighting, or safety) nodes/things are tailored to

a specific task. It is therefore unlikely that one single manufacturer will create all nodes in a building. Hence, interoperability as well as trust bootstrapping between nodes of different vendors is important.

The thing is later installed and commissioned within a network by an installer during the bootstrapping phase. Specifically, the device identity and the secret keys used during normal operation may be provided to the device during this phase. Different subcontractors may install different IoT devices for different purposes. Furthermore, the installation and bootstrapping procedures may not be a discrete event and may stretch over an extended period. After being bootstrapped, the device and the system of things are in operational mode and execute the functions of the BAC system. During this operational phase, the device is under the control of the system owner and used by multiple system users. For devices with lifetimes spanning several years, occasional maintenance cycles may be required. During each maintenance phase, the software on the device can be upgraded or applications running on the device can be reconfigured. The maintenance tasks can be performed either locally or from a backend system. Depending on the operational changes to the device, it may be required to re-bootstrap at the end of a maintenance cycle. The device continues to loop through the operational phase and the eventual maintenance phases until the device is decommissioned at the end of its lifecycle. However, the end-of-life of a device does not necessarily mean that it is defective and rather denotes a need to replace and upgrade the network to the next-generation devices for additional functionality. Therefore, the device can be removed and re-commissioned to be used in a different system under a different owner thereby starting the lifecycle all over again.

```
    _Manufactured           _SW update          _Decommissioned
   /                       /                   /
   |   _Installed          |   _ Application   |   _Removed &
   |  /                    |  / reconfigured   |  /  replaced
   |  |   _Commissioned    |  |                |  |
   |  |  /                 |  |                |  |   _Reownership &
   |  |  |    _Application |  |   _Application |  |  / recommissioned
   |  |  |   /   running   |  |  / running     |  |  |
   |  |  |   |             |  |  |             |  |  |             \\
   +##+##+###+#############+##+##+#############+##+##+##############>>>
       \/  \______________/ \/  \_____________/ \___/         time //
       /           /         \          \          \
   Bootstrapping  /      Maintenance &   \     Maintenance &
                 /      re-bootstrapping  \   re-bootstrapping
           Operational                Operational
```

Figure 1: The lifecycle of a thing in the Internet of Things

2.2. Security building blocks

Security is a key requirement in any communication system. However, security is an even more critical requirement in real-world IoT deployments for several reasons. First, compromised IoT systems can not only endanger the privacy and security of a user, but can also cause physical harm. This is because IoT systems often comprise sensors, actuators and other connected devices in the physical environment of the user which could adversely affect the user if they are compromised. Second, a vulnerable IoT system means that an attacker can alter the functionality of a device from a given manufacturer. This not only affects the manufacturer's brand image, but can also leak information that is very valuable for the manufacturer (such as proprietary algorithms). Third, the impact of attacking an IoT system goes beyond a specific device or an isolated system since compromised IoT systems can be misused at scale. For example, they may be used to perform a Distributed Denial of Service (DDoS) attack that limits the availability of other networks and services. The fact that many IoT systems rely on standard IP protocols allows for easier system integration, but this also makes standard attacks applicable to a wide number of devices deployed in multiple systems. This results in new requirements regarding the implementation of security.

The term security subsumes a wide range of primitives, protocols, and procedures. Firstly, it includes the basic provision of security services that include confidentiality, authentication, integrity, authorization, source authentication, and availability along with some augmented services, such as duplicate detection and detection of stale packets (timeliness). These security services can be

implemented by means of a combination of cryptographic mechanisms, such as block ciphers, hash functions, or signature algorithms, and non-cryptographic mechanisms, which implement authorization and other security policy enforcement aspects. For ensuring security in IoT networks, we should not only focus on the required security services, but also pay special attention to how these services are realized in the overall system and how the security functionalities are executed in practice. To this end, we consider five major "building blocks" to analyze and classify security aspects for IoT:

1. IoT security architecture: refers to the system-level elements involved in the management of security relationships between things (for example, centralized or distributed). For instance, a smart home could rely on a centralized key distribution center in charge of managing cryptographic keys, devices, users, access control and privacy policies.

2. The security model within a thing: describes the way security parameters, keys, processes, and applications are managed within a smart object. This includes aspects such as application process separation, secure storage of key materials, etc. For instance, some smart objects might have extremely limited resources and limited capabilities to protect secret keys. In contrast, other devices used in critical applications, such as a pacemaker, may rely on methods to protect cryptographic keys and functionality.

3. Secure bootstrapping: denotes the process by which a thing securely joins an IoT system at a given location and point of time. For instance, bootstrapping of a connected camera can include the authentication and authorization of the device as well as the transfer of security parameters necessary for operation in a given network.

4. Network security: describes the mechanisms applied within a network to ensure secure operation. Specifically, it prevents attackers from endangering or modifying the expected operation of a smart object. It also protects the network itself from malicious things. Network security can include several mechanisms ranging from data link layer security, secure routing, and network layer security.

5. Application security: describes mechanisms to allow secure transfer of application data. The security may be implemented at different layers of the Internet protocol suite. For instance, assume a smart object such as an environmental sensor that is connected to a backend system. Application security here can refer to the exchange of secure blocks of data such as

measurements between the sensor and the backed, or it can also refer to a software update for the smart object. This data is exchanged end-to-end independently of the underlying network infrastructure, for example through proxies or other store-and-forward mechanisms.

```
               ..........................
               :           +-----------+:
               :       *+*>|Application|*****
               :       *|  +-----------+:   *
               :       *|  +-----------+:   *
               :       *|->| Transport |:   *
               :    * _*|  +-----------+:   *
               :    *|  |  +-----------+:   *
               :    *|  |->|  Network  |:   *
               :    *|  |  +-----------+:   *
               :    *|  |  +-----------+:   *    *** Bootstrapping
               :    *|  +->|     L2    |:   *    ~~~ Transport Security
               :    *|     +-----------+:   *    ''' Object Security
               :+--------+              :   *
               :|Security| Configuration:   *
               :|Service |   Entity     :   *
               :+--------+              :   *
               :........................:   *
                                            *
   .........................                *  .........................
   :+--------+             :                *  :             +--------+:
   :|Security|   Node B    :                *  :   Node A    |Security|:
   :|Service |             :                *  :             |Service |:
   :+--------+             :                *  :             +--------+:
   :    |     +-----------+:                *  :+-----------+     |*   :
   :    |  +->|Application|:                ****|Application|<*+* |*   :
   :    |  |  +-----------+:''''''''''''''''''''+-----------+  |* |*   :
   :    |  |  +-----------+:                   :+-----------+  |* |*   :
   :    |  |->| Transport |~~~~~~~~~~~~~~~~~~~~~| Transport |<-|* |*   :
   :    |__|  +-----------+: ................. :+-----------+  |*_|*   :
   :       |  +-----------+: : +-----------+ : :+-----------+  | *     :
   :       |->|  Network  |: : |  Network  | : :|  Network  |<-|       :
   :       |  +-----------+: : +-----------+ : :+-----------+  |       :
   :       |  +-----------+: : +-----------+ : :+-----------+  |       :
   :       +->|     L2    |: : |     L2    | : :|     L2    |<-+       :
   :          +-----------+: : +-----------+ : :+-----------+          :
   :.......................: :...............: :.......................:
```

Figure 2: Overview of Security Mechanisms

Inspired by the security framework for routing over low power and lossy network [RFC7416], we show an example security model of a smart

object and illustrate how different security concepts and lifecycle phases map to the Internet communication stack. Assume a centralized architecture in which a configuration entity stores and manages the identities of the things associated with the system along with their cryptographic keys. During the bootstrapping phase, each thing executes the bootstrapping protocol with the configuration entity, thus obtaining the required device identities and the keying material. The security service on a thing in turn stores the received keying material for the network layer and application security mechanisms for secure communication. Things can then securely communicate with each other during their operational phase by means of the employed network and application security mechanisms.

3. Security Threats and Managing Risk

Security threats in related IP protocols have been analyzed in multiple documents including HTTPS [RFC2818], COAP [RFC7252], 6LoWPAN [RFC4919], ANCP [RFC5713], DNS security threats [RFC3833], IPv6 ND [RFC3756], and PANA [RFC4016]. In this section, we specifically discuss the threats that could compromise an individual thing, or the network as a whole. Note that these set of threats might go beyond the scope of Internet protocols but we gather them here for the sake of completeness. We also note that these threats can be classified according to either (i) the thing's lifecycle phases (when does the threat occur?) or (ii) the security building blocks (which functionality is affected by the threat?):

1. Cloning of things: During the manufacturing process of a thing, an untrusted factory can easily clone the physical characteristics, firmware/software, or security configuration of the thing. Deployed things might also be compromised and their software reverse engineered allowing for cloning or software modifications. Such a cloned thing may be sold at a cheaper price in the market, and yet can function normally as a genuine thing. For example, two cloned devices can still be associated and work with each other. In the worst-case scenario, a cloned device can be used to control a genuine device or perform an attack. One should note here, that an untrusted factory may also change functionality of the cloned thing, resulting in degraded functionality with respect to the genuine thing (thereby, inflicting potential damage to the reputation of the original thing manufacturer). Moreover, additional functionality can be introduced in the cloned thing, an example of such functionality is a backdoor.

2. Malicious substitution of things: During the installation of a thing, a genuine thing may be substituted with a similar variant (of lower quality) without being detected. The main motivation

may be cost savings, where the installation of lower-quality things (for example, non-certified products) may significantly reduce the installation and operational costs. The installers can subsequently resell the genuine things to gain further financial benefits. Another motivation may be to inflict damage to the reputation of a competitor's offerings.

3. Eavesdropping attack: During the commissioning of a thing into a network, it may be susceptible to eavesdropping, especially if operational keying materials, security parameters, or configuration settings, are exchanged in clear using a wireless medium or if used cryptographic algorithms are not suitable for the envisioned lifetime of the device and the system. After obtaining the keying material, the attacker might be able to recover the secret keys established between the communicating entities, thereby compromising the authenticity and confidentiality of the communication channel, as well as the authenticity of commands and other traffic exchanged over this communication channel. When the network is in operation, T2T communication may be eavesdropped upon if the communication channel is not sufficiently protected or in the event of session key compromise due to protocol weaknesses or a long period of usage without key renewal or updates. Messages can also be recorded and processed offline at a later time.

4. Man-in-the-middle attack: Both the commissioning phase and operational phases may also be vulnerable to man-in-the-middle attacks, for example, when keying material between communicating entities is exchanged in the clear and the security of the key establishment protocol depends on the tacit assumption that no third party can eavesdrop during the execution of this protocol. Additionally, device authentication or device authorization may be non-trivial, or may need support of a human decision process, since things usually do not have a-priori knowledge about each other and cannot always differentiate friends and foes via completely automated mechanisms. Thus, even if the key establishment protocol provides cryptographic device authentication, this knowledge on device identities may still need complementing with a human-assisted authorization step (thereby, presenting a weak link and offering the potential of man-in-the-middle attacks this way).

5. Firmware attacks: When a thing is in operation or maintenance phase, its firmware or software may be updated to allow for new functionality or new features. An attacker may be able to exploit such a firmware upgrade by replacing the thing's software with malicious software, thereby influencing the operational behavior of the thing. For example, an attacker

could add a piece of malicious code to the firmware that will cause it to periodically report the energy usage of the lamp to a data repository for analysis. Similarly, devices whose software has not been properly maintained and updated might contain vulnerabilities that might be exploited by attackers to replace the firmware on the device.

6. Extraction of private information: IoT devices (such as sensors, actuators, etc.) are often physically unprotected in their ambient environment and they could easily be captured by an attacker. An attacker with physical access may then attempt to extract private information such as keys (for example, device's key, private-key, group key), sensed data (for example, healthcare status of a user), configuration parameters (for example, the Wi-Fi key), or proprietary algorithms (for example, algorithm performing some data analytics task). Even when the data originating from a thing is encrypted, attackers can perform traffic analysis to deduce meaningful information which might compromise the privacy of the thing's owner and/or user.

7. Routing attack: As highlighted in [ID-Daniel], routing information in IoT can be spoofed, altered, or replayed, in order to create routing loops, attract/repel network traffic, extend/shorten source routes, etc. Other relevant routing attacks include 1) Sinkhole attack (or blackhole attack), where an attacker declares himself to have a high-quality route/path to the base station, thus allowing him to do manipulate all packets passing through it. 2) Selective forwarding, where an attacker may selectively forward packets or simply drop a packet. 3) Wormhole attack, where an attacker may record packets at one location in the network and tunnel them to another location, thereby influencing perceived network behavior and potentially distorting statistics, thus greatly impacting the functionality of routing. 4) Sybil attack, whereby an attacker presents multiple identities to other things in the network.

8. Elevation of privilege: An attacker with low privileges can misuse additional flaws in the implemented authentication and authorization mechanisms of a thing to gain more privileged access to the thing and its data.

9. Privacy threat: The tracking of a thing's location and usage may pose a privacy risk to its users. For instance, an attacker can infer information based on the information gathered about individual things, thus deducing behavioral patterns of the user of interest to him. Such information may subsequently be sold to interested parties for marketing purposes and targeted advertising. In extreme cases, such information might be used

to track dissidents in oppressive regimes. Unlawful surveillance and interception of traffic to/from a thing by intelligence agencies is also a privacy threat.

10. Denial-of-Service (DoS) attack: Often things have very limited memory and computation capabilities. Therefore, they are vulnerable to resource exhaustion attack. Attackers can continuously send requests to specific things so as to deplete their resources. This is especially dangerous in the Internet of Things since an attacker might be located in the backend and target resource-constrained devices that are part of a constrained node network [RFC7228]. DoS attack can also be launched by physically jamming the communication channel. Network availability can also be disrupted by flooding the network with a large number of packets. On the other hand, things compromised by attackers can be used to disrupt the operation of other networks or systems by means of a Distributed DoS (DDoS) attack.

To deal with above threats it is required to find and apply suitable security mitigations. However, new threats and exploits appear on a daily basis and products are deployed in different environments prone to different types of threats. Thus, ensuring a proper level of security in an IoT system at any point of time is challenging. To address this challenge, a process for secure product creation is required to ensure that an IoT system is secure and no security risks are present. A non-exhaustive list of required methodologies include:

1. A Business Impact Analysis (BIA) assesses the consequences of the loss of basic security attributes: confidentiality, integrity and availability in an IoT system. These consequences might include the impact from lost data, reduced sales, increased expenses, regulatory fines, customer dissatisfaction, etc. Performing a business impact analysis allows a business to determine the relevance of having a proper security design.

2. A Risk Assessment (RA) analyzes security threats to an IoT system while considering their likelihood and impact. It also includes categorizing each of them with a risk level. Risks classified as moderate or high must be mitigated, i.e., the security architecture should be able to deal with those threat.

3. A privacy impact assessment (PIA) aims at assessing the Personally Identifiable Information (PII) that is collected, processed, or used in an IoT system. By doing so, the goal is to fulfill applicable legal requirements, determine risks and effects of manipulation and loss of PII.

4. Procedures for incident reporting and mitigation refer to the methodologies that allow becoming aware of any security issues that affect an IoT system. Furthermore, this includes steps towards the actual deployment of patches that mitigate the identified vulnerabilities.

BIA, RA, and PIA should generally be realized during the creation of a new IoT system or when deploying significant system/feature upgrades. In general, it is recommended to re-assess them on a regular basis taking into account new use cases and/or threats.

4. State-of-the-Art

This section is organized as follows. Section 4.1 summarizes state-of-the-art on IP-based IoT systems, within IETF and in other standardization bodies. Section 4.2 summarizes state-of-the-art on IP-based security protocols and their usage. Section 4.3 discusses guidelines and regulations for securing IoT as proposed by other bodies.

4.1. IP-based IoT Protocols and Standards

Nowadays, there exists a multitude of control protocols for IoT. For BAC systems, the ZigBee standard [ZB], BACNet [BACNET], and DALI [DALI] play key roles. Recent trends, however, focus on an all-IP approach for system control.

In this setting, a number of IETF working groups are designing new protocols for resource-constrained networks of smart things. The 6LoWPAN working group [WG-6LoWPAN] for example has defined methods and protocols for the efficient transmission and adaptation of IPv6 packets over IEEE 802.15.4 networks [RFC4944].

The CoRE working group [WG-CoRE] has specified the Constrained Application Protocol (CoAP) [RFC7252]. CoAP is a RESTful protocol for constrained devices that is modeled after HTTP and typically runs over UDP to enable efficient application-level communication for things.

In many smart object networks, the smart objects are dispersed and have intermittent reachability either because of network outages or because they sleep during their operational phase to save energy. In such scenarios, direct discovery of resources hosted on the constrained server might not be possible. To overcome this barrier, the CoRE working group is specifying the concept of a Resource Directory (RD) [ID-rd]. The Resource Directory hosts descriptions of resources which are located on other nodes. These resource descriptions are specified as CoRE link format [RFC6690].

While CoAP defines a standard communication protocol, a format for representing sensor measurements and parameters over CoAP is required. The Sensor Measurement Lists (SenML) [ID-senml] is a specification that defines media types for simple sensor measurements and parameters. It has a minimalistic design so that constrained devices with limited computational capabilities can easily encode their measurements and, at the same time, servers can efficiently collect large number of measurements.

In many IoT deployments, the resource-constrained smart objects are connected to the Internet via a gateway that is directly reachable. For example, an IEEE 802.11 Access Point (AP) typically connects the client devices to the Internet over just one wireless hop. However, some deployments of smart object networks require routing between the smart objects themselves. The IETF has therefore defined the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) [RFC6550]. RPL provides support for multipoint-to-point traffic from resource-constrained smart objects towards a more resourceful central control point, as well as point-to-multipoint traffic in the reverse direction. It also supports point-to-point traffic between the resource-constrained devices. A set of routing metrics and constraints for path calculation in RPL are also specified [RFC6551].

The IPv6 over Networks of Resource-constrained Nodes (6lo) [WG-6lo] working group of the IETF has specified how IPv6 packets can be transmitted over various link layer protocols that are commonly employed for resource-constrained smart object networks. There is also ongoing work to specify IPv6 connectivity for a Non-Broadcast Multi-Access (NBMA) mesh network that is formed by IEEE 802.15.4 TimeSlotted Channel Hopping (TSCH} links [ID-6tisch]. Other link layer protocols for which IETF has specified or is currently specifying IPv6 support include Bluetooth [RFC7668], Digital Enhanced Cordless Telecommunications (DECT) Ultra Low Energy (ULE) air interface [RFC8105], and Near Field Communication (NFC) [ID-6lonfc].

Baker and Meyer [RFC6272] identify which IP protocols can be used in smart grid environments. They give advice to smart grid network designers on how they can decide on a profile of the Internet protocol suite for smart grid networks.

JavaScript Object Notation (JSON) is a lightweight text representation format for structured data [RFC7159]. It is often used for transmitting serialized structured data over the network. IETF has defined specifications for encoding cryptographic keys, encrypted content, signed content, and claims to be transferred between two parties as JSON objects. They are referred to as JSON Web Keys (JWK) [RFC7517], JSON Web Encryption (JWE) [RFC7516], JSON Web Signatures (JWS) [RFC7515] and JSON Web Token (JWT) [RFC7519].

An alternative to JSON, Concise Binary Object Representation (CBOR) [RFC7049] is a concise binary data format that is used for serialization of structured data. It is designed for resource-constrained nodes and therefore it aims to provide a fairly small message size with minimal implementation code, and extensibility without the need for version negotiation. CBOR Object Signing and Encryption (COSE) [RFC8152] specifies how to encode cryptographic keys, message authentication codes, encrypted content, and signatures with CBOR.

The Light-Weight Implementation Guidance (LWIG) working group [WG-LWIG] is collecting experiences from implementers of IP stacks in constrained devices. The working group has already produced documents such as RFC7815 [RFC7815] which defines how a minimal Internet Key Exchange Version 2 (IKEv2) initiator can be implemented.

The Thing-2-Thing Research Group (T2TRG) [RG-T2TRG] is investigating the remaining research issues that need to be addressed to quickly turn the vision of IoT into a reality where resource-constrained nodes can communicate with each other and with other more capable nodes on the Internet.

Additionally, industry alliances and other standardization bodies are creating constrained IP protocol stacks based on the IETF work. Some important examples of this include:

1. Thread [Thread]: Specifies the Thread protocol that is intended for a variety of IoT devices. It is an IPv6-based network protocol that runs over IEEE 802.15.4.

2. Industrial Internet Consortium [IIoT]: The consortium defines reference architectures and security frameworks for development, adoption and widespread use of Industrial Internet technologies based on existing IETF standards.

3. Internet Protocol for Smart Objects IPSO [IPSO]: The alliance specifies a common object model that enables application software on any device to interoperate with other conforming devices.

4. OneM2M [OneM2M]: The standards body defines technical and API specifications for IoT devices. It aims to create a service layer that can run on any IoT device hardware and software.

5. Open Connectivity Foundation (OCF) [OCF]: The foundation develops standards and certifications primarily for IoT devices that use Constrained Application Protocol (CoAP) as the application layer protocol.

6. Fairhair Alliance [Fairhair]: Specifies an IoT middleware to enable interoperability between different application standards used in building automation and lighting systems.

7. OMA LWM2M [LWM2M]: OMA Lightweight M2M is a standard from the Open Mobile Alliance for M2M and IoT device management. LWM2M relies on CoAP as the application layer protocol and uses a RESTful architecture for remote management of IoT devices.

4.2. Existing IP-based Security Protocols and Solutions

There are three main security objectives for IoT networks: 1. protecting the IoT network from attackers. 2. protecting IoT applications and thus, the things and users. 3. protecting the rest of the Internet and other things from attacks that use compromised things as an attack platform.

In the context of the IP-based IoT deployments, consideration of existing Internet security protocols is important. There are a wide range of specialized as well as general-purpose security solutions for the Internet domain such as IKEv2/IPsec [RFC7296], TLS [RFC5246], DTLS [RFC6347], HIP [RFC7401], PANA [RFC5191], and EAP [RFC3748].

There is ongoing work to define an authorization and access-control framework for resource-constrained nodes. The Authentication and Authorization for Constrained Environments (ACE) [WG-ACE] working group is defining a solution to allow only authorized access to resources that are hosted on a smart object server and are identified by a URI. The current proposal [ID-aceoauth] is based on the OAuth 2.0 framework [RFC6749] and it comes with profiles intended for different communication scenarios, e.g. DTLS Profile for Authentication and Authorization for Constrained Environments [ID-acedtls].

The CoAP base specification [RFC7252] provides a description of how DTLS can be used for securing CoAP. It proposes three different modes for using DTLS: the PreSharedKey mode, where nodes have pre-provisioned keys for initiating a DTLS session with another node, RawPublicKey mode, where nodes have asymmetric-key pairs but no certificates to verify the ownership, and Certificate mode, where public keys are certified by a certification authority. An IoT implementation profile [RFC7925] is defined for TLS version 1.2 and DTLS version 1.2 that offers communication security for resource-constrained nodes.

Transport Layer Security (TLS) and its datagram-oriented variant DTLS secure transport-layer connections. TLS provides security for TCP and requires a reliable transport, while DTLS secures and uses

datagram-oriented protocols such as UDP. Both protocols are intentionally kept similar and share the same ideology and cipher suites.

OSCOAP [ID-OSCOAP] is a proposal that protects CoAP messages by wrapping them in the CBOR Object Signing and Encryption (COSE) [RFC8152] format. Thus, OSCOAP falls in the category of object security and it can be applied wherever CoAP can be used. The advantage of OSCOAP over DTLS is that it provides some more flexibility when dealing with end-to-end security. Section 5.1.3 discusses this further.

The Automated Certificate Management Environment (ACME) [WG-ACME] working group is specifying conventions for automated X.509 certificate management. This includes automatic validation of certificate issuance, certificate renewal, and certificate revocation. While the initial focus of working group is on domain name certificates (as used by web servers), other uses in some IoT deployments is possible.

The Internet Key Exchange (IKEv2)/IPsec and the Host Identity protocol (HIP) reside at or above the network layer in the OSI model. Both protocols are able to perform an authenticated key exchange and set up the IPsec for secure payload delivery. Currently, there are also ongoing efforts to create a HIP variant coined Diet HIP [ID-HIP-DEX] that takes constrained networks and nodes into account at the authentication and key exchange level.

Migault et al. [ID-dietesp] are working on a compressed version of IPsec so that it can easily be used by resource-constrained IoT devices. They rely on the Internet Key Exchange Protocol version 2 (IKEv2) for negotiating the compression format.

The Extensible Authentication Protocol (EAP) [RFC3748] is an authentication framework supporting multiple authentication methods. EAP runs directly over the data link layer and, thus, does not require the deployment of IP. It supports duplicate detection and retransmission, but does not allow for packet fragmentation. The Protocol for Carrying Authentication for Network Access (PANA) is a network-layer transport for EAP that enables network access authentication between clients and the network infrastructure. In EAP terms, PANA is a UDP-based EAP lower layer that runs between the EAP peer and the EAP authenticator.

Figure 3 depicts the relationships between the discussed protocols in the context of the security terminology introduced in Section 2.

```
             ..........................
             :           +-----------+:
             :       *+*>|Application|*****     *** Bootstrapping
             :       *|  +-----------+:   *     ### Transport Security
             :       *|  +-----------+:   *     === Network security
             :       *|->| Transport |:   *     ''' Object security
             :    * _*|  +-----------+:   *
             :    *|  |  +-----------+:   *
             :    *|  |->|  Network  |:   *--> -PANA/EAP
             :    *|  |  +-----------+:   *    -HIP
             :    *|  |  +-----------+:   *
             :    *|  +->|     L2    |:   *     ## DTLS
             :    *|     +-----------+:   *     '' OSCOAP
             :+--------+              :   *
             :|Security| Configuration:   *     [] HIP,IKEv2
             :|Service |   Entity     :   *     [] ESP/AH
             :+--------+              :   *
             :........................:   *
                                          *
 .........................                *    .........................
 :+--------+             :                *    :             +--------+:
 :|Security|   Node B    :    Secure      *    :   Node A    |Security|:
 :|Service |             :    routing     *    :             |Service |:
 :+--------+             :   framework    *    :             +--------+:
 :    |     +-----------+:        |       **** :+-----------+     |*   :
 :    |  +->|Application|:''''''''|'''''''''''':|Application|<*+* |*   :
 :    |  |  +----##-----+:        |            :+----##-----+  |* |*   :
 :    |  |  +----##-----+:        |            :+----##-----+  |* |*   :
 :    |  |->| Transport |#########|#############| Transport |<-|* |*   :
 :    |__|  +----[]-----+:  ......|..........  :+----[]-----+  |*_|*   :
 :       |  +====[]=====+=====+===========+=====+====[]=====+  | *     :
 :       |->|| Network  |:  : |  Network  | :  :|  Network ||<-|       :
 :       |  +|----------+:  : +-----------+ :  :+----------|+  |       :
 :       |  +|----------+:  : +-----------+ :  :+----------|+  |       :
 :       +->||    L2    |:  : |     L2    | :  :|     L2   ||<-+       :
 :          +===========+=====+===========+=====+===========+         :
 :.......................:  :..............:  :......................:
```

Figure 3: Relationships between IP-based security protocols

4.3. IoT Security Guidelines

Attacks on and from IoT devices have become common in the last years, for instance, large scale Denial of Service (DoS) attacks on the Internet Infrastructure from compromised IoT devices. This fact has prompted many different standards bodies and consortia to provide guidelines for developers and the Internet community at large to

build secure IoT devices and services. A subset of the different guidelines and ongoing projects are as follows:

1. GSMA IoT security guidelines [GSMAsecurity]: GSMA has published a set of security guidelines for the benefit of new IoT product and service providers. The guidelines are aimed at device manufacturers, service providers, developers and network operators. An enterprise can complete an IoT Security Self-Assessment to demonstrate that its products and services are aligned with the security guidelines of the GSMA.

2. BITAG Internet of Things (IoT) Security and Privacy Recommendations [BITAG]: Broadband Internet Technical Advisory Group (BITAG) has also published recommendations for ensuring security and privacy of IoT device users. BITAG observes that many IoT devices are shipped from the factory with software that is already outdated and vulnerable. The report also states that many devices with vulnerabilities will not be fixed either because the manufacturer does not provide updates or because the user does not apply them. The recommendations include that IoT devices should function without cloud and Internet connectivity, and that all IoT devices should have methods for automatic secure software updates.

3. CSA New Security Guidance for Early Adopters of the IoT [CSA]: The Cloud Security Alliance (CSA) recommendations for early adopters of IoT encourages enterprises to implement security at different layers of the protocol stack. It also recommends implementation of an authentication/authorization framework for IoT deployments. A complete list of recommendations is available in the report [CSA].

4. U.S. Department of Homeland Security [DHS]: DHS has put forth six strategic principles that would enable IoT developers, manufacturers, service providers and consumers to maintain security as they develop, manufacture, implement or use network-connected IoT devices.

5. NIST [NIST-Guide]: The NIST special publication urges enterprise and US federal agencies to address security throughout the systems engineering process. The publication builds upon the ISO/IEC 15288 standard and augments each process in the system lifecycle with security enhancements.

6. NIST [nist_lightweight_project]: NIST is running a project on lightweight cryptography with the purpose of: (i) identifying application areas for which standard cryptographic algorithms are too heavy, classifying them according to some application

profiles to be determined; (ii) determining limitations in those existing cryptographic standards; and (iii) standardizing lightweight algorithms that can be used in specific application profiles.

7. OWASP [OWASP]: Open Web Application Security Project (OWASP) provides security guidance for IoT manufactures, developers and consumers. OWASP also includes guidelines for those who intend to test and analyze IoT devices and applications.

8. IoT Security foundation [IoTSecFoundation]: IoT security foundation has published a document that enlists various considerations that need to be taken into account when developing IoT applications. For example, the document states that IoT devices could use hardware-root of trust to ensure that only authorized software runs on the devices.

9. NHTSA [NHTSA]: The US National Highway Traffic Safety Administration provides a set of non-binding guidance to the automotive industry for improving the cyber security of vehicles. While some of the guidelines are general, the document provides specific recommendations for the automotive industry such as how various automotive manufacturer can share cyber security vulnerabilities discovered.

10. Best Current Practices (BCP) for IoT devices [ID-Moore]: This document provides a list of minimum requirements that vendors of Internet of Things (IoT) devices should to take into account while developing applications, services and firmware updates in order to reduce the frequency and severity of security incidents that arise from compromised IoT devices.

11. ENISA [ENISA_ICS]: The European Union Agency for Network and Information Security published a document on communication network dependencies for ICS/SCADA systems in which security vulnerabilities, guidelines and general recommendations are summarized.

Other guideline and recommendation documents may exist or may later be published. This list should be considered non-exhaustive. Despite the acknowledgment that security in the Internet is needed and the existence of multiple guidelines, the fact is that many IoT devices and systems have very limited security. There are multiple reasons for this. For instance, some manufactures focus on delivering a product without paying enough attention to security. This may be because of lack of expertise or limited budget. However, the deployment of such insecure devices poses a severe threat on the privacy and safety of users. The vast amount of devices and their

inherent mobile nature also implies that an initially secure system can become insecure if a compromised device gains access to the system at some point in time. Even if all other devices in a given environment are secure, this does not prevent external (passive) attacks caused by insecure devices.

Recently the Federal Communications Commission (FCC) [FCC] has stated the need for additional regulation of IoT systems. FCC identifies this as a missing component, especially for Federal Information Systems (FIS). Today, security in the US FIS is regulated according to Federal Information Security Management Act (FISMA). From this law, NIST has derived a number of new documents to categorize FIS and determine minimum security requirements for each category. These minimum security requirements are specified in NIST SP 800-53r4 [NIST-SP80053].

Even with strong regulations in place, the question remains as to how such regulations can be applied in practice to non-federal deployments, such as industrial, homes, offices, or smart cities. Each of them exhibits unique features, involves very diverse types of users, has different operational requirements, and combines IoT devices from multiple manufacturers. Future regulations should therefore consider such diverse deployment scenarios.

5. Challenges for a Secure IoT

In this section, we take a closer look at the various security challenges in the operational and technical features of IoT and then discuss how existing Internet security protocols cope with these technical and conceptual challenges through the lifecycle of a thing. This discussion should neither be understood as a comprehensive evaluation of all protocols, nor can it cover all possible aspects of IoT security. Yet, it aims at showing concrete limitations and challenges in some IoT design areas rather than giving an abstract discussion. In this regard, the discussion handles issues that are most important from the authors' perspectives.

5.1. Constraints and Heterogeneous Communication

Coupling resource-constrained networks and the powerful Internet is a challenge because the resulting heterogeneity of both networks complicates protocol design and system operation. In the following we briefly discuss the resource constraints of IoT devices and the consequences for the use of Internet Protocols in the IoT domain.

5.1.1. Resource Constraints

IoT deployments are often characterized by lossy and low-bandwidth communication channels. IoT devices are also often constrained in terms of CPU, memory, and energy budget available [RFC7228]. These characteristics directly impact the threats to and the design of security protocols for the IoT domain. First, the use of small packets, for example, IEEE 802.15.4 supports 127-byte sized packets at the physical layer, may result in fragmentation of larger packets required by security protocols. This may open new attack vectors for state exhaustion DoS attacks, which is especially tragic, for example, if the fragmentation is caused by large key exchange messages of security protocols. Moreover, packet fragmentation commonly downgrades the overall system performance due to fragment losses and the need for retransmissions. For instance, fate-sharing packet flight as implemented by DTLS might aggravate the resulting performance loss.

The size and number of messages should be minimized to reduce memory requirements and optimize bandwidth usage. In this context, layered approaches involving a number of protocols might lead to worse performance in resource-constrained devices since they combine the headers of the different protocols. In some settings, protocol negotiation can increase the number of exchanged messages. To improve performance during basic procedures such as, for example, bootstrapping, it might be a good strategy to perform those procedures at a lower layer.

Small CPUs and scarce memory limit the usage of resource-expensive cryptographic primitives such as public-key cryptography as used in most Internet security standards. This is especially true if the basic cryptographic blocks need to be frequently used or the underlying application demands low delay.

Independently from the development in the IoT domain, all discussed security protocols show efforts to reduce the cryptographic cost of the required public-key-based key exchanges and signatures with Elliptic Curve Cryptography (ECC) [RFC5246], [RFC5903], [RFC7401], and [ID-HIP-DEX]. Moreover, all protocols have been revised in the last years to enable cryptographic agility, making cryptographic primitives interchangeable. However, these improvements are only a first step in reducing the computation and communication overhead of Internet protocols. The question remains if other approaches can be applied to leverage key agreement in these heavily resource-constrained environments.

A further fundamental need refers to the limited energy budget available to IoT nodes. Careful protocol (re)design and usage is

required to reduce not only the energy consumption during normal operation, but also under DoS attacks. Since the energy consumption of IoT devices differs from other device classes, judgments on the energy consumption of a particular protocol cannot be made without tailor-made IoT implementations.

5.1.2. Denial-of-Service Resistance

The tight memory and processing constraints of things naturally alleviate resource exhaustion attacks. Especially in unattended T2T communication, such attacks are difficult to notice before the service becomes unavailable (for example, because of battery or memory exhaustion). As a DoS countermeasure, DTLS, IKEv2, HIP, and Diet HIP implement return routability checks based on a cookie mechanism to delay the establishment of state at the responding host until the address of the initiating host is verified. The effectiveness of these defenses strongly depend on the routing topology of the network. Return routability checks are particularly effective if hosts cannot receive packets addressed to other hosts and if IP addresses present meaningful information as is the case in today's Internet. However, they are less effective in broadcast media or when attackers can influence the routing and addressing of hosts (for example, if hosts contribute to the routing infrastructure in ad-hoc networks and meshes).

In addition, HIP implements a puzzle mechanism that can force the initiator of a connection (and potential attacker) to solve cryptographic puzzles with variable difficulties. Puzzle-based defense mechanisms are less dependent on the network topology but perform poorly if CPU resources in the network are heterogeneous (for example, if a powerful Internet host attacks a thing). Increasing the puzzle difficulty under attack conditions can easily lead to situations where a powerful attacker can still solve the puzzle while weak IoT clients cannot and are excluded from communicating with the victim. Still, puzzle-based approaches are a viable option for sheltering IoT devices against unintended overload caused by misconfiguration or malfunctioning things.

5.1.3. End-to-end security, protocol translation, and the role of middleboxes

The term end-to-end security often has multiple interpretations. Here, we consider end-to-end security in the context end-to-end IP connectivity, from a sender to a receiver. Services such as confidentiality and integrity protection on packet data, message authentication codes or encryption are typically used to provide end-to-end security. These protection methods render the protected parts of the packets immutable as rewriting is either not possible because

a) the relevant information is encrypted and inaccessible to the gateway or b) rewriting integrity-protected parts of the packet would invalidate the end-to-end integrity protection.

Protocols for constrained IoT networks are not exactly identical to their larger Internet counterparts for efficiency and performance reasons. Hence, more or less subtle differences between protocols for constrained IoT networks and Internet protocols will remain. While these differences can be bridged with protocol translators at middleboxes, they may become major obstacles if end-to-end security measures between IoT devices and Internet hosts are needed.

If access to data or messages by the middleboxes is required or acceptable, then a diverse set of approaches for handling such a scenario are available. Note that some of these approaches affect the meaning of end-to-end security in terms of integrity and confidentiality since the middleboxes will be able to either decrypt or modify partially the exchanged messages:

1. Sharing credentials with middleboxes enables them to transform (for example, decompress, convert, etc.) packets and re-apply the security measures after transformation. This method abandons end-to-end security and is only applicable to simple scenarios with a rudimentary security model.

2. Reusing the Internet wire format for IoT makes conversion between IoT and Internet protocols unnecessary. However, it can lead to poor performance in some use cases because IoT specific optimizations (for example, stateful or stateless compression) are not possible.

3. Selectively protecting vital and immutable packet parts with a message authentication code or with encryption requires a careful balance between performance and security. Otherwise this approach might either result in poor performance or poor security depending on which parts are selected for protection, where they are located in the original packet, and how they are processed. [ID-OSCOAP] proposes a solution in this direction by encrypting and integrity protecting most of the message fields except those parts that a middlebox needs to read or change.

4. Homomorphic encryption techniques can be used in the middlebox to perform certain operations. However, this is limited to data processing involving arithmetic operations. Furthermore, performance of existing libraries, for example, SEAL [SEAL] is still too limited and it is is not widely applicable yet.

5. Message authentication codes that sustain transformation can be realized by considering the order of transformation and protection (for example, by creating a signature before compression so that the gateway can decompress the packet without recalculating the signature). Such an approach enables IoT specific optimizations but is more complex and may require application-specific transformations before security is applied. Moreover, the usage of encrypted or integrity-protected data prevents middleboxes from transforming packets.

6. Mechanisms based on object security can bridge the protocol worlds, but still require that the two worlds use the same object security formats. Currently the object security format based on CBOR Object Signing and Encryption (COSE) [RFC8152] (IoT protocol) is different from JSON Object Signing and Encryption (JOSE) [RFC7520] or Cryptographic Message Syntax (CMS) [RFC5652]. Legacy devices relying on traditional Internet protocols will need to update to the newer protocols for constrained environments to enable real end-to-end security. Furthermore, middleboxes do not have any access to the data and this approach does not prevent an attacker from modifying relevant fields in CoAP.

To the best of our knowledge, none of the mentioned security approaches that focus on the confidentiality and integrity of the communication exchange between two IP end-points provide the perfect solution in this problem space.

We finally note that end-to-end security can also be considered in the context of availability: making sure that the messages are delivered. In this case, the end-points cannot control this, but the middleboxes play a fundamental role to make sure that exchanged messages are not dropped, for example, due to a DDoS attack.

5.1.4. New network architectures and paradigm

There is a multitude of new link layer protocols that aim to address the resource-constrained nature of IoT devices. For example, the IEEE 802.11 ah [IEEE802ah] has been specified for extended range and lower energy consumption to support Internet of Things (IoT) devices. Similarly, Low-Power Wide-Area Network (LPWAN) protocols such as LoRa [lora], Sigfox [sigfox], NarrowBand IoT (NB-IoT) [nbiot] are all designed for resource-constrained devices that require long range and low bit rates. While these protocols allow IoT devices to conserve energy and operate efficiently, they also add additional security challenges. For example, the relatively small MTU can make security handshakes with large X509 certificates a significant overhead. At the same time, new communication paradigms also allow IoT devices to

communicate directly amongst themselves with or without support from the network. This communication paradigm is also referred to as Device-to-Device (D2D) or Machine-to-Machine (M2M) or Thing-to-Thing (T2T) communication and it is motivated by a number of features such as improved network performance, lower latency and lower energy requirements.

5.2. Bootstrapping of a Security Domain

Creating a security domain from a set of previously unassociated IoT devices is a key operation in the lifecycle of a thing in an IoT network. This aspect is further elaborated and discussed in the T2TRG draft on bootstrapping [ID-bootstrap].

5.3. Operational Challenges

After the bootstrapping phase, the system enters the operational phase. During the operational phase, things can use the state information created during the bootstrapping phase in order to exchange information securely. In this section, we discuss the security challenges during the operational phase. Note that many of the challenges discussed in Section 5.1 apply during the operational phase.

5.3.1. Group Membership and Security

Group key negotiation is an important security service for IoT communication patterns in which a thing sends some data to multiple things or data flows from multiple things towards a thing. All discussed protocols only cover unicast communication and therefore, do not focus on group-key establishment. This applies in particular to (D)TLS and IKEv2. Thus, a solution is required in this area. A potential solution might be to use the Diffie-Hellman keys - that are used in IKEv2 and HIP to setup a secure unicast link - for group Diffie-Hellman key-negotiations. However, Diffie-Hellman is a relatively heavy solution, especially if the group is large.

Symmetric and asymmetric keys can be used in group communication. Asymmetric keys have the advantage that they can provide source authentication. However, doing broadcast encryption with a single public/private key pair is also not feasible. Although a single symmetric key can be used to encrypt the communication or compute a message authentication code, it has inherent risks since the capture of a single node can compromise the key shared throughout the network. The usage of symmetric-keys also does not provide source authentication. Another factor to consider is that asymmetric cryptography is more resource-intensive than symmetric key solutions. Thus, the security risks and performance trade-offs of applying

either symmetric or asymmetric keys to a given IoT use case need to be well-analyzed according to risk and usability assessments. [ID-multicast] is looking at a combination of symmetric (for encryption) and asymmetric (for authentication) in the same packet.

Conceptually, solutions that provide secure group communication at the network layer (IPsec/IKEv2, HIP/Diet HIP) may have an advantage in terms of the cryptographic overhead when compared to application-focused security solutions (TLS/ DTLS). This is due to the fact that application-focused solutions require cryptographic operations per group application, whereas network layer approaches may allow sharing secure group associations between multiple applications (for example, for neighbor discovery and routing or service discovery). Hence, implementing shared features lower in the communication stack can avoid redundant security measures. However, it is important to note that sharing security contexts among different applications involves potential security threats, e.g., if one of the applications is malicious and monitors exchanged messages or injects fake messages. In the case of OSCOAP, it provides security for CoAP group communication as defined in RFC7390, i.e., based on multicast IP. If the same security association is reused for each application, then this solution does not seem to have more cryptographic overhead compared to IPsec.

Several group key solutions have been developed by the MSEC working group [WG-MSEC] of the IETF. The MIKEY architecture [RFC4738] is one example. While these solutions are specifically tailored for multicast and group broadcast applications in the Internet, they should also be considered as candidate solutions for group key agreement in IoT. The MIKEY architecture for example describes a coordinator entity that disseminates symmetric keys over pair-wise end-to-end secured channels. However, such a centralized approach may not be applicable in a distributed IoT environment, where the choice of one or several coordinators and the management of the group key is not trivial.

5.3.2. Mobility and IP Network Dynamics

It is expected that many things (for example, wearable sensors, and user devices) will be mobile in the sense that they are attached to different networks during the lifetime of a security association. Built-in mobility signaling can greatly reduce the overhead of the cryptographic protocols because unnecessary and costly re-establishments of the session (possibly including handshake and key agreement) can be avoided. IKEv2 supports host mobility with the MOBIKE [RFC4555] and [RFC4621] extension. MOBIKE refrains from applying heavyweight cryptographic extensions for mobility. However, MOBIKE mandates the use of IPsec tunnel mode which requires to

transmit an additional IP header in each packet. This additional overhead could be alleviated by using header compression methods or the Bound End- to-End Tunnel (BEET) mode [ID-Nikander], a hybrid of tunnel and transport mode with smaller packet headers.

HIP offers a simple yet effective mobility management by allowing hosts to signal changes to their associations [RFC8046]. However, slight adjustments might be necessary to reduce the cryptographic costs, for example, by making the public-key signatures in the mobility messages optional. Diet HIP does not define mobility yet but it is sufficiently similar to HIP and can use the same mechanisms. TLS and DTLS do not have native mobility support, however, work on DTLS mobility exists in the form of an Internet draft [ID-Williams]. The specific need for IP-layer mobility mainly depends on the scenario in which the nodes operate. In many cases, mobility supported by means of a mobile gateway may suffice to enable mobile IoT networks, such as body sensor networks. Using message based application-layer security solutions such as OSCoAP [ID-OSCOAP] can also alleviate the problem of re-establishing lower-layer sessions for mobile nodes.

5.4. Secure software update and cryptographic agility

IoT devices are often expected to stay functional for several years and decades even though they might operate unattended with direct Internet connectivity. Software updates for IoT devices are therefore not only required for new functionality, but also to eliminate security vulnerabilities due to software bugs, design flaws, or deprecated algorithms. Software bugs might remain even after careful code review. Implementations of security protocols might contain (design) flaws. Cryptographic algorithms can also become insecure due to advances in cryptanalysis.

Schneier [SchneierSecurity] in his essay highlights several challenges that hinder mechanisms for secure software update of IoT devices. First, there is a lack of incentives for manufactures, vendors and others on the supply chain to issue updates for their devices. Second, parts of the software running on IoT devices is simply a binary blob without any source code available. Since the complete source code is not available, no patches can be written for that piece of code. Lastly Schneier points out that even when updates are available, users generally have to manually download and install them. However, users are never alerted about security updates and at many times do not have the necessary expertise to manually administer the required updates.

The FTC staff report on Internet of Things - Privacy & Security in a Connected World [FTCreport] and the Article 29 Working Party Opinion

8/2014 on the Recent Developments on the Internet of Things [Article29] also document the challenges for secure remote software update of IoT devices. They note that even providing such a software update capability may add new vulnerabilities for constrained devices. For example, a buffer overflow vulnerability in the implementation of a software update protocol (TR69) [TR69] and an expired certificate in a hub device [wink] demonstrate how the software update process itself can introduce vulnerabilities.

Powerful IoT devices that run general purpose operating systems can make use of sophisticated software update mechanisms known from the desktop world. However, resource-constrained devices typically do not have any operating system and are often not equipped with a memory management unit or similar tools. Therefore, they might require more specialized solutions.

An important requirement for secure software and firmware updates is source authentication. Source authentication requires the resource-constrained things to implement public-key signature verification algorithms. As stated in Section 5.1.1, resource-constrained things have limited amount of computational capabilities and energy supply available which can hinder the amount and frequency of cryptographic processing that they can perform. In addition to source authentication, software updates might require confidential delivery over a secure (encrypted) channel. The complexity of broadcast encryption can force the usage of point-to-point secure links - however, this increases the duration of a software update in a large system. Alternatively, it may force the usage of solutions in which the software update is delivered to a gateway, and then distributed to the rest of the system with a network key. Sending large amounts of data that later needs to be assembled and verified over a secure channel can consume a lot of energy and computational resources. Correct scheduling of the software updates is also a crucial design challenge. For example, a user of connected light bulbs would not want them to update and restart at night. More importantly, the user would not want all the lights to update at the same time.

Software updates in IoT systems are also needed to update old and insecure cryptographic primitives. However, many IoT systems, some of which are already deployed, are not designed with provisions for cryptographic agility. For example, many devices come with a wireless radio that has an AES128 hardware co-processor. These devices solely rely on the co-processor for encrypting and authenticating messages. A software update adding support for new cryptographic algorithms implemented solely in software might not fit on these devices due to limited memory, or might drastically hinder its operational performance. This can lead to the use of old and insecure devices. Therefore, it is important to account for the fact

that cryptographic algorithms would need to be updated and consider the following when planning for cryptographic agility:

1. Would it be safe to use the existing cryptographic algorithms available on the device for updating with new cryptographic algorithms that are more secure?

2. Will the new software-based implementation fit on the device given the limited resources?

3. Would the normal operation of existing IoT applications on the device be severely hindered by the update?

Finally, we would like to highlight the previous and ongoing work in the area of secure software and firmware updates at the IETF. [RFC4108] describes how Cryptographic Message Syntax (CMS) [RFC5652] can be used to protect firmware packages. The IAB has also organized a workshop to understand the challenges for secure software update of IoT devices. A summary of the workshop and the proposed next steps have been documented [iotsu]. Finally, a new working group called Software Updates for Internet of Things (suit) [WG-SUIT] is currently being chartered at the IETF. The working group aims to standardize a new version [RFC4108] that reflects the best current practices for firmware update based on experience with IoT deployments. It will specifically work on describing an IoT firmware update architecture and specifying a manifest format that contains meta-data about the firmware update package.

5.5. End-of-Life

Like all commercial devices, IoT devices have a given useful lifetime. The term end-of-life (EOL) is used by vendors or network operators to indicate the point of time in which they limit or end support for the IoT product. This may be planned or unplanned (for example when the vendor or manufacturer goes bankrupt or when a network operator moves to a different type of networking technology). A user should still be able to use and perhaps even update the device. This requires for some form of authorization handover.

Although this may seem far-fetched given the commercial interests and market dynamics, we have examples from the mobile world where the devices have been functional and up-to-date long after the original vendor stopped supporting the device. CyanogenMod for Android devices, and OpenWrt for home routers are two such instances where users have been able to use and update their devices even after the official EOL. Admittedly it is not easy for an average user to install and configure their devices on their own. With the deployment of millions of IoT devices, simpler mechanisms are needed

to allow users to add new root-of-trusts and install software and firmware from other sources once the device is EOL.

5.6. Verifying device behavior

Users using new IoT appliances such as Internet-connected smart televisions, speakers and cameras are often unaware that these devices can undermine their privacy. Recent revelations have shown that many IoT device vendors have been collecting sensitive private data through these connected appliances with or without appropriate user warnings [cctv].

An IoT device user/owner would like to monitor and verify its operational behavior. For instance, the user might want to know if the device is connecting to the server of the manufacturer for any reason. This feature - connected to the manufacturer's server - may be necessary in some scenarios, such as during the initial configuration of the device. However, the user should be kept aware of the data that the device is sending back to the vendor. For example, the user might want to know if his/her TV is sending data when he/she inserts a new USB stick.

Providing such information to the users in an understandable fashion is challenging. This is because IoT devices are not only resource-constrained in terms of their computational capability, but also in terms of the user interface available. Also, the network infrastructure where these devices are deployed will vary significantly from one user environment to another. Therefore, where and how this monitoring feature is implemented still remains an open question.

Manufacturer Usage Description (MUD) files [ID-MUD] are perhaps a first step towards implementation of such a monitoring service. The idea behind MUD files is relatively simple: IoT devices would disclose the location of their MUD file to the network during installation. The network can then retrieve those files, and learn about the intended behavior of the devices stated by the device manufacturer. A network monitoring service could then warn the user/owner of devices if they don't behave as expected.

5.7. Testing: bug hunting and vulnerabilities

Given that IoT devices often have inadvertent vulnerabilities, both users and developers would want to perform extensive testing on their IoT devices, networks, and systems. Nonetheless, since the devices are resource-constrained and manufactured by multiple vendors, some of them very small, devices might be shipped with very limited

testing, so that bugs can remain and can be exploited at a later stage. This leads to two main types of challenges:

1. It remains to be seen how the software testing and quality assurance mechanisms used from the desktop and mobile world will be applied to IoT devices to give end users the confidence that the purchased devices are robust.

2. It is also an open question how the combination of devices from multiple vendors might actually lead to dangerous network configurations, for example, if combination of specific devices can trigger unexpected behavior.

5.8. Quantum-resistance

Many IoT systems that are being deployed today will remain operational for many years. With the advancements made in the field of quantum computers, it is possible that large-scale quantum computers are available in the future for performing cryptanalysis on existing cryptographic algorithms and cipher suites. If this happens, it will have two consequences. First, functionalities enabled by means of RSA/ECC - namely key exchange, public-key encryption and signature - would not be secure anymore due to Shor's algorithm. Second, the security level of symmetric algorithms will decrease, for example, the security of a block cipher with a key size of b bits will only offer b/2 bits of security due to Grover's algorithm.

The above scenario becomes more urgent when we consider the so called "harvest and decrypt" attack in which an attacker can start to harvest (store) encrypted data today, before a quantum-computer is available, and decrypt it years later, once a quantum computer is available.

This situation would require us to move to quantum-resistant alternatives, in particular, for those functionalities involving key exchange, public-key encryption and signatures. [ID-c2pq] describes when quantum computers may become widely available and what steps are necessary for transition to cryptographic algorithms that provide security even in presence of quantum computers. While future planning is hard, it may be a necessity in certain critical IoT deployments which are expected to last decades or more. Although increasing the key-size of the different algorithms is definitely an option, it would also incur additional computational overhead and network traffic. This would be undesirable in most scenarios. There have been recent advancements in quantum-resistant cryptography.

We refer to [ETSI_GR_QSC_001] for an extensive overview of existing quantum-resistant cryptography. [RFC7696] provides guidelines for cryptographic algorithm agility.

5.9. Privacy protection

Users will be surrounded by hundreds of connected IoT devices. Even if the communication links are encrypted and protected, information about the users might be collected for different purposes affecting their privacy. In [Ziegeldorf], privacy in IoT is defined as the threefold guarantee to the user for: 1. awareness of privacy risks imposed by smart things and services surrounding the data subject, 2. individual control over the collection and processing of personal information by the surrounding smart things, 3. awareness and control of subsequent use and dissemination of personal information by those entities to any entity outside the subject's personal control sphere.

Based on this definition, several privacy threats and challenges have been documented [Ziegeldorf] and [RFC6973]:

1. Identification - refers to the identification of the users and their objects.

2. Localization - relates to the capability of locating a user and even tracking them.

3. Profiling - is about creating a profile of the user and their preferences.

4. Interaction - occurs when a user has been profiled and a given interaction is preferred, presenting (for example, visually) some information that discloses private information.

5. Lifecycle transitions - take place when devices are, for example, sold without properly removing private data.

6. Inventory attacks - happen if specific information about (smart) objects in possession of a user is disclosed.

7. Linkage - is about when information of two of more IoT systems is combined so that a broader view on the personal data is created.

When IoT systems are deployed, the above issues should be considered to ensure that private data remains private. These issues are particularly challenging in environments in which multiple users with different privacy preferences interact with the same IoT devices. For example, an IoT device controlled by user A (low privacy settings) might leak private information about another user B (high

privacy settings). How to deal with these threats in practice is an area of ongoing research.

5.10. Data leakage

Many IoT devices are resource-constrained and often deployed in unattended environments. Some of these devices can also be purchased off-the-shelf or online without any credential-provisioning process. Therefore, an attacker can have direct access to the device and apply advanced techniques to retrieve information that a traditional black box model does not consider. Example of those techniques are side-channel attacks or code disassembly. By doing this, the attacker can try to retrieve data such as:

1. long term keys. These long term keys can be extracted by means of a side-channel attack or reverse engineering. If these keys are exposed, then they might be used to perform attacks on devices deployed in other locations.

2. source code that might allow the attacker to determine bugs or find exploits to perform other types of attacks. The attacker might also just sell the source code.

3. proprietary algorithms. The attacker can analyze these algorithms gaining valuable know-how. The attacker can also create copies of the product (based on those proprietary algorithms) or modify the algorithms to perform more advanced attacks.

Protection against such data leakage patterns is not trivial since devices are inherently resource-constrained. An open question is whether there are any viable techniques to protect IoT devices and the data in the devices in such an adversarial model.

5.11. Trustworthy IoT Operation

Flaws in the design and implementation of a secure IoT device and network can lead to security vulnerabilities. For instance, a flaw is the distribution of an Internet-connected IoT device in which a default password is used in all devices. Many IoT devices can be found in the Internet by means of tools such as Shodan [shodan], and if they have any vulnerability, it can then be exploited at scale, for example, to launch DDoS attacks. For instance, Dyn, a major DNS, was attacked by means of a DDoS attack originated from a large IoT botnet composed of thousands of compromised IP-cameras [dyn-attack]. Open questions in this area are:

1. How to prevent large scale vulnerabilities in IoT devices?

2. How to prevent attackers from exploiting vulnerabilities in IoT devices at large scale?

3. If the vulnerability has been exploited, how do we stop a large scale attack before any damage is caused?

Some ideas are being explored to address this issue. One of this approaches refers to the specification of Manufacturer Usage Description (MUD) files [ID-MUD]. As explained earlier, this proposal requires IoT devices to disclose the location of their MUD file to the network during installation. The network can then (i) retrieve those files, (ii) learn from the manufacturers the intended usage of the devices, for example, which services they require to access, and then (iii) create suitable filters such as firewall rules.

6. Conclusions and Next Steps

This Internet Draft provides IoT security researchers, system designers and implementers with an overview of both operational and security requirements in the IP-based Internet of Things. We discuss a general threat model, threats, state-of-the-art, and security challenges.

Although plenty of steps have been realized during the last few years (summarized in Section 4.1) and many organizations are publishing general recommendations (Section 4.3) describing how IoT should be secured, there are many challenges ahead that require further attention. Challenges of particular importance are bootstrapping of security, group security, secure software updates, long-term security and quantum-resistance, privacy protection, data leakage prevention - where data could be cryptographic keys, personal data, or even algorithms - and ensuring trustworthy IoT operation. All these problems are important; however, different deployment environments have different operational and security demands. Thus, a potential approach is the definition and standardization of security profiles, each with specific mitigation strategies according to the risk assessment associated with the security profile. Such an approach would ensure minimum security capabilities in different environments while ensuring interoperability.

7. Security Considerations

This document reflects upon the requirements and challenges of the security architectural framework for the Internet of Things.

8. IANA Considerations

This document contains no request to IANA.

9. Acknowledgments

We gratefully acknowledge feedback and fruitful discussion with Tobias Heer, Robert Moskowitz, Thorsten Dahm, Hannes Tschofenig, Carsten Bormann, Barry Raveendran, Ari Keranen, Goran Selander, Fred Baker and Eliot Lear. We acknowledge the additional authors of the previous version of this document Sye Loong Keoh, Rene Hummen and Rene Struik.

10. Informative References

[Article29]
"Opinion 8/2014 on the on Recent Developments on the Internet of Things", Web http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2014/wp223_en.pdf, n.d..

[AUTO-ID] "AUTO-ID LABS", Web http://www.autoidlabs.org/, September 2010.

[BACNET] "BACnet", Web http://www.bacnet.org/, February 2011.

[BITAG] "Internet of Things (IoT) Security and Privacy Recommendations", Web http://www.bitag.org/report-internet-of-things-security-privacy-recommendations.php, n.d..

[cctv] "Backdoor In MVPower DVR Firmware Sends CCTV Stills To an Email Address In China", Web https://hardware.slashdot.org/story/16/02/17/0422259/backdoor-in-mvpower-dvr-firmware-sends-cctv-stills-to-an-email-address-in-china, n.d..

[CSA] "Security Guidance for Early Adopters of the Internet of Things (IoT)", Web https://downloads.cloudsecurityalliance.org/whitepapers/Security_Guidance_for_Early_Adopters_of_the_Internet_of_Things.pdf, n.d..

[DALI] "DALI", Web http://www.dalibydesign.us/dali.html, February 2011.

[DHS] "Strategic Principles For Securing the Internet of Things (IoT)", Web https://www.dhs.gov/sites/default/files/publications/Strategic_Principles_for_Securing_the_Internet_of_Things-2016-1115-FINAL....pdf, n.d..

[dyn-attack] "Dyn Analysis Summary Of Friday October 21 Attack", Web https://dyn.com/blog/dyn-analysis-summary-of-friday-october-21-attack/, n.d..

[ENISA_ICS] "Communication network dependencies for ICS/SCADA Systems", European Union Agency For Network And Information Security , February 2017.

[ETSI_GR_QSC_001] "Quantum-Safe Cryptography (QSC);Quantum-safe algorithmic framework", European Telecommunications Standards Institute (ETSI) , June 2016.

[Fairhair] "Fairhair Alliance", Web https://www.fairhair-alliance.org/, n.d..

[FCC] "Federal Communications Comssion Response 12-05-2016", FCC , February 2016.

[FTCreport] "FTC Report on Internet of Things Urges Companies to Adopt Best Practices to Address Consumer Privacy and Security Risks", Web https://www.ftc.gov/news-events/press-releases/2015/01/ftc-report-internet-things-urges-companies-adopt-best-practices, n.d..

[GSMAsecurity] "GSMA IoT Security Guidelines", Web http://www.gsma.com/connectedliving/future-iot-networks/iot-security-guidelines/, n.d..

[ID-6lonfc] Choi, Y., Hong, Y., Youn, J., Kim, D., and J. Choi, "Transmission of IPv6 Packets over Near Field Communication", draft-ietf-6lo-nfc-08 (work in progress), October 2017.

[ID-6tisch] Thubert, P., "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4", draft-ietf-6tisch-architecture-13 (work in progress), November 2017.

[ID-acedtls] Gerdes, S., Bergmann, O., Bormann, C., Selander, G., and L. Seitz, "Datagram Transport Layer Security (DTLS) Profiles for Authentication and Authorization for Constrained Environments (ACE)", draft-ietf-ace-dtls-authorize-02 (work in progress), October 2017.

[ID-aceoauth] Seitz, L., Selander, G., Wahlstroem, E., Erdtman, S., and H. Tschofenig, "Authentication and Authorization for Constrained Environments (ACE)", draft-ietf-ace-oauth-authz-09 (work in progress), November 2017.

[ID-bootstrap] Sarikaya, B., Sethi, M., and A. Sangi, "Secure IoT Bootstrapping: A Survey", draft-sarikaya-t2trg-sbootstrapping-03 (work in progress), February 2017.

[ID-c2pq] Hoffman, P., "The Transition from Classical to Post-Quantum Cryptography", draft-hoffman-c2pq-02 (work in progress), August 2017.

[ID-Daniel] Park, S., Kim, K., Haddad, W., Chakrabarti, S., and J. Laganier, "IPv6 over Low Power WPAN Security Analysis", draft-daniel-6lowpan-security-analysis-05 (work in progress), March 2011.

[ID-dietesp] Migault, D., Guggemos, T., and C. Bormann, "Diet-ESP: a flexible and compressed format for IPsec/ESP", draft-mglt-6lo-diet-esp-02 (work in progress), July 2016.

[ID-HIP-DEX] Moskowitz, R., "HIP Diet EXchange (DEX)", draft-moskowitz-hip-rg-dex-06 (work in progress), May 2012.

[ID-Moore] Moore, K., Barnes, R., and H. Tschofenig, "Best Current Practices for Securing Internet of Things (IoT) Devices", draft-moore-iot-security-bcp-01 (work in progress), July 2017.

[ID-MUD] Lear, E., Droms, R., and D. Romascanu, "Manufacturer Usage Description Specification", draft-ietf-opsawg-mud-13 (work in progress), October 2017.

[ID-multicast] Tiloca, M., Selander, G., Palombini, F., and J. Park, "Secure group communication for CoAP", draft-tiloca-core-multicast-oscoap-04 (work in progress), October 2017.

[ID-Nikander] Nikander, P. and J. Melen, "A Bound End-to-End Tunnel (BEET) mode for ESP", draft-nikander-esp-beet-mode-09 (work in progress), August 2008.

[ID-OSCOAP] Selander, G., Mattsson, J., Palombini, F., and L. Seitz, "Object Security for Constrained RESTful Environments (OSCORE)", draft-ietf-core-object-security-07 (work in progress), November 2017.

[ID-rd] Shelby, Z., Koster, M., Bormann, C., Stok, P., and C. Amsuess, "CoRE Resource Directory", draft-ietf-core-resource-directory-12 (work in progress), October 2017.

[ID-senml] Jennings, C., Shelby, Z., Arkko, J., Keranen, A., and C. Bormann, "Media Types for Sensor Measurement Lists (SenML)", draft-ietf-core-senml-11 (work in progress), October 2017.

[ID-Williams] Williams, M. and J. Barrett, "Mobile DTLS", draft-barrett-mobile-dtls-00 (work in progress), March 2009.

[IEEE802ah] "Status of Project IEEE 802.11ah, IEEE P802.11- Task Group AH-Meeting Update.", Web http://www.ieee802.org/11/Reports/tgah_update.htm, n.d..

[IIoT] "Industrial Internet Consortium", Web http://www.iiconsortium.org/, n.d..

[IoTSecFoundation] "Establishing Principles for Internet of Things Security", Web https://iotsecurityfoundation.org/establishing-principles-for-internet-of-things-security/, n.d..

[iotsu] "Patching the Internet of Things: IoT Software Update Workshop 2016", Web https://www.ietf.org/blog/2016/07/patching-the-internet-of-things-iot-software-update-workshop-2016/, n.d..

[IPSO] "IPSO Alliance", Web http://www.ipso-alliance.org, n.d..

[lora] "LoRa - Wide Area Networks for IoT", Web https://www.lora-alliance.org/, n.d..

[LWM2M] "OMA LWM2M", Web http://openmobilealliance.org/iot/lightweight-m2m-lwm2m, n.d..

[nbiot] "NarrowBand IoT", Web http://www.3gpp.org/ftp/tsg_ran/TSG_RAN/TSGR_69/Docs/RP-151621.zip, n.d..

[NHTSA] "Cybersecurity Best Practices for Modern Vehicles", Web https://www.nhtsa.gov/staticfiles/nvs/pdf/812333_CybersecurityForModernVehicles.pdf, n.d..

[NIST-Guide] Ross, R., McEvilley, M., and J. Oren, "Systems Security Engineering", Web http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-160.pdf, n.d..

[NIST-SP80053] "Security and Privacy Controls for Federal Information Systems and Organizations", Web http://dx.doi.org/10.6028/NIST.SP.800-53r4, n.d..

[nist_lightweight_project] "NIST lightweight Project", Web www.nist.gov/programs-projects/lightweight-cryptography, www.nist.gov/sites/default/files/documents/2016/10/17/sonmez-turan-presentation-lwc2016.pdf, n.d..

[nist_pq] "NIST Post-Quantum-Cryptography Project", Web https://csrc.nist.gov/Projects/Post-Quantum-Cryptography, n.d..

[OCF] "Open Connectivity Foundation", Web https://openconnectivity.org/, n.d..

[OneM2M] "OneM2M", Web http://www.onem2m.org/, n.d..

[OWASP] "IoT Security Guidance", Web https://www.owasp.org/index.php/IoT_Security_Guidance, n.d..

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, DOI 10.17487/RFC2818, May 2000, <https://www.rfc-editor.org/info/rfc2818>.

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

[RFC3756] Nikander, P., Ed., Kempf, J., and E. Nordmark, "IPv6 Neighbor Discovery (ND) Trust Models and Threats", RFC 3756, DOI 10.17487/RFC3756, May 2004, <https://www.rfc-editor.org/info/rfc3756>.

[RFC3833] Atkins, D. and R. Austein, "Threat Analysis of the Domain Name System (DNS)", RFC 3833, DOI 10.17487/RFC3833, August 2004, <https://www.rfc-editor.org/info/rfc3833>.

[RFC4016] Parthasarathy, M., "Protocol for Carrying Authentication and Network Access (PANA) Threat Analysis and Security Requirements", RFC 4016, DOI 10.17487/RFC4016, March 2005, <https://www.rfc-editor.org/info/rfc4016>.

[RFC4108] Housley, R., "Using Cryptographic Message Syntax (CMS) to Protect Firmware Packages", RFC 4108, DOI 10.17487/RFC4108, August 2005, <https://www.rfc-editor.org/info/rfc4108>.

[RFC4555] Eronen, P., "IKEv2 Mobility and Multihoming Protocol (MOBIKE)", RFC 4555, DOI 10.17487/RFC4555, June 2006, <https://www.rfc-editor.org/info/rfc4555>.

[RFC4621] Kivinen, T. and H. Tschofenig, "Design of the IKEv2 Mobility and Multihoming (MOBIKE) Protocol", RFC 4621, DOI 10.17487/RFC4621, August 2006, <https://www.rfc-editor.org/info/rfc4621>.

[RFC4738] Ignjatic, D., Dondeti, L., Audet, F., and P. Lin, "MIKEY-RSA-R: An Additional Mode of Key Distribution in Multimedia Internet KEYing (MIKEY)", RFC 4738, DOI 10.17487/RFC4738, November 2006, <https://www.rfc-editor.org/info/rfc4738>.

[RFC4919] Kushalnagar, N., Montenegro, G., and C. Schumacher, "IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals", RFC 4919, DOI 10.17487/RFC4919, August 2007, <https://www.rfc-editor.org/info/rfc4919>.

[RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler, "Transmission of IPv6 Packets over IEEE 802.15.4 Networks", RFC 4944, DOI 10.17487/RFC4944, September 2007, <https://www.rfc-editor.org/info/rfc4944>.

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

[RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <https://www.rfc-editor.org/info/rfc5246>.

[RFC5652] Housley, R., "Cryptographic Message Syntax (CMS)", STD 70, RFC 5652, DOI 10.17487/RFC5652, September 2009, <https://www.rfc-editor.org/info/rfc5652>.

[RFC5713] Moustafa, H., Tschofenig, H., and S. De Cnodder, "Security Threats and Security Requirements for the Access Node Control Protocol (ANCP)", RFC 5713, DOI 10.17487/RFC5713, January 2010, <https://www.rfc-editor.org/info/rfc5713>.

[RFC5903] Fu, D. and J. Solinas, "Elliptic Curve Groups modulo a Prime (ECP Groups) for IKE and IKEv2", RFC 5903, DOI 10.17487/RFC5903, June 2010, <https://www.rfc-editor.org/info/rfc5903>.

[RFC6272] Baker, F. and D. Meyer, "Internet Protocols for the Smart Grid", RFC 6272, DOI 10.17487/RFC6272, June 2011, <https://www.rfc-editor.org/info/rfc6272>.

[RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, DOI 10.17487/RFC6347, January 2012, <https://www.rfc-editor.org/info/rfc6347>.

[RFC6550] Winter, T., Ed., Thubert, P., Ed., Brandt, A., Hui, J., Kelsey, R., Levis, P., Pister, K., Struik, R., Vasseur, JP., and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks", RFC 6550, DOI 10.17487/RFC6550, March 2012, <https://www.rfc-editor.org/info/rfc6550>.

[RFC6551] Vasseur, JP., Ed., Kim, M., Ed., Pister, K., Dejean, N., and D. Barthel, "Routing Metrics Used for Path Calculation in Low-Power and Lossy Networks", RFC 6551, DOI 10.17487/RFC6551, March 2012, <https://www.rfc-editor.org/info/rfc6551>.

[RFC6568] Kim, E., Kaspar, D., and JP. Vasseur, "Design and Application Spaces for IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs)", RFC 6568, DOI 10.17487/RFC6568, April 2012, <https://www.rfc-editor.org/info/rfc6568>.

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

[RFC6749] Hardt, D., Ed., "The OAuth 2.0 Authorization Framework", RFC 6749, DOI 10.17487/RFC6749, October 2012, <https://www.rfc-editor.org/info/rfc6749>.

[RFC6973] Cooper, A., Tschofenig, H., Aboba, B., Peterson, J., Morris, J., Hansen, M., and R. Smith, "Privacy Considerations for Internet Protocols", RFC 6973, DOI 10.17487/RFC6973, July 2013, <https://www.rfc-editor.org/info/rfc6973>.

[RFC7049] Bormann, C. and P. Hoffman, "Concise Binary Object Representation (CBOR)", RFC 7049, DOI 10.17487/RFC7049, October 2013, <https://www.rfc-editor.org/info/rfc7049>.

[RFC7159] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", RFC 7159, DOI 10.17487/RFC7159, March 2014, <https://www.rfc-editor.org/info/rfc7159>.

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

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

[RFC7296] Kaufman, C., Hoffman, P., Nir, Y., Eronen, P., and T. Kivinen, "Internet Key Exchange Protocol Version 2 (IKEv2)", STD 79, RFC 7296, DOI 10.17487/RFC7296, October 2014, <https://www.rfc-editor.org/info/rfc7296>.

[RFC7401] Moskowitz, R., Ed., Heer, T., Jokela, P., and T. Henderson, "Host Identity Protocol Version 2 (HIPv2)", RFC 7401, DOI 10.17487/RFC7401, April 2015, <https://www.rfc-editor.org/info/rfc7401>.

[RFC7416] Tsao, T., Alexander, R., Dohler, M., Daza, V., Lozano, A., and M. Richardson, Ed., "A Security Threat Analysis for the Routing Protocol for Low-Power and Lossy Networks (RPLs)", RFC 7416, DOI 10.17487/RFC7416, January 2015, <https://www.rfc-editor.org/info/rfc7416>.

[RFC7515] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Signature (JWS)", RFC 7515, DOI 10.17487/RFC7515, May 2015, <https://www.rfc-editor.org/info/rfc7515>.

[RFC7516] Jones, M. and J. Hildebrand, "JSON Web Encryption (JWE)", RFC 7516, DOI 10.17487/RFC7516, May 2015, <https://www.rfc-editor.org/info/rfc7516>.

[RFC7517] Jones, M., "JSON Web Key (JWK)", RFC 7517, DOI 10.17487/RFC7517, May 2015, <https://www.rfc-editor.org/info/rfc7517>.

[RFC7519] Jones, M., Bradley, J., and N. Sakimura, "JSON Web Token (JWT)", RFC 7519, DOI 10.17487/RFC7519, May 2015, <https://www.rfc-editor.org/info/rfc7519>.

[RFC7520] Miller, M., "Examples of Protecting Content Using JSON Object Signing and Encryption (JOSE)", RFC 7520, DOI 10.17487/RFC7520, May 2015, <https://www.rfc-editor.org/info/rfc7520>.

[RFC7668] Nieminen, J., Savolainen, T., Isomaki, M., Patil, B., Shelby, Z., and C. Gomez, "IPv6 over BLUETOOTH(R) Low Energy", RFC 7668, DOI 10.17487/RFC7668, October 2015, <https://www.rfc-editor.org/info/rfc7668>.

[RFC7696] Housley, R., "Guidelines for Cryptographic Algorithm Agility and Selecting Mandatory-to-Implement Algorithms", BCP 201, RFC 7696, DOI 10.17487/RFC7696, November 2015, <https://www.rfc-editor.org/info/rfc7696>.

[RFC7744] Seitz, L., Ed., Gerdes, S., Ed., Selander, G., Mani, M., and S. Kumar, "Use Cases for Authentication and Authorization in Constrained Environments", RFC 7744, DOI 10.17487/RFC7744, January 2016, <https://www.rfc-editor.org/info/rfc7744>.

[RFC7815] Kivinen, T., "Minimal Internet Key Exchange Version 2 (IKEv2) Initiator Implementation", RFC 7815, DOI 10.17487/RFC7815, March 2016, <https://www.rfc-editor.org/info/rfc7815>.

[RFC7925] Tschofenig, H., Ed. and T. Fossati, "Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", RFC 7925, DOI 10.17487/RFC7925, July 2016, <https://www.rfc-editor.org/info/rfc7925>.

[RFC8046] Henderson, T., Ed., Vogt, C., and J. Arkko, "Host Mobility with the Host Identity Protocol", RFC 8046, DOI 10.17487/RFC8046, February 2017, <https://www.rfc-editor.org/info/rfc8046>.

[RFC8105] Mariager, P., Petersen, J., Ed., Shelby, Z., Van de Logt, M., and D. Barthel, "Transmission of IPv6 Packets over Digital Enhanced Cordless Telecommunications (DECT) Ultra Low Energy (ULE)", RFC 8105, DOI 10.17487/RFC8105, May 2017, <https://www.rfc-editor.org/info/rfc8105>.

[RFC8152] Schaad, J., "CBOR Object Signing and Encryption (COSE)", RFC 8152, DOI 10.17487/RFC8152, July 2017, <https://www.rfc-editor.org/info/rfc8152>.

[RG-T2TRG]
"IRTF Thing-to-Thing (T2TRG) Research Group", Web https://datatracker.ietf.org/rg/t2trg/charter/, n.d..

[SchneierSecurity]
"The Internet of Things Is Wildly Insecure--And Often Unpatchable", Web https://www.schneier.com/essays/archives/2014/01/the_internet_of_thin.html, n.d..

[SEAL] "Simple Encrypted Arithmetic Library - SEAL", Web https://sealcrypto.codeplex.com/, n.d..

[shodan] "Shodan", Web https://www.shodan.io/, n.d..

[sigfox] "Sigfox - The Global Communications Service Provider for the Internet of Things (IoT)", Web https://www.sigfox.com/, n.d..

[Thread] "Thread Group", Web http://threadgroup.org/, n.d..

[TR69] "Too Many Cooks - Exploiting the Internet-of-TR-069-Things", Web https://media.ccc.de/v/31c3_-_6166_-_en_-_saal_6_-_201412282145_-_too_many_cooks_-_exploiting_the_internet-of-tr-069-things_-_lior_oppenheim_-_shahar_tal, n.d..

[WG-6lo] "IETF IPv6 over Networks of Resource-constrained Nodes (6lo) Working Group", Web https://datatracker.ietf.org/wg/6lo/charter/, n.d..

[WG-6LoWPAN] "IETF IPv6 over Low power WPAN (6lowpan) Working Group", Web http://tools.ietf.org/wg/6lowpan/, n.d..

[WG-ACE] "IETF Authentication and Authorization for Constrained Environments (ACE) Working Group", Web https://datatracker.ietf.org/wg/ace/charter/, n.d..

[WG-ACME] "Automated Certificate Management Environment Working Group", Web https://datatracker.ietf.org/wg/acme/about/, n.d..

[WG-CoRE] "IETF Constrained RESTful Environment (CoRE) Working Group", Web https://datatracker.ietf.org/wg/core/charter/, n.d..

[WG-LWIG] "IETF Light-Weight Implementation Guidance (LWIG) Working Group", Web https://datatracker.ietf.org/wg/lwig/charter/, n.d..

[WG-MSEC] "IETF MSEC Working Group", Web https://datatracker.ietf.org/wg/msec/, n.d..

[WG-SUIT] "IETF Software Updates for Internet of Things (suit)", Web https://datatracker.ietf.org/group/suit/about/, n.d..

[wink] "Wink's Outage Shows Us How Frustrating Smart Homes Could Be", Web http://www.wired.com/2015/04/smart-home-headaches/, n.d..

[ZB] "ZigBee Alliance", Web http://www.zigbee.org/, February 2011.

[Ziegeldorf] Ziegeldorf, J., Garcia-Morchon, O., and K. Wehrle,, "Privacy in the Internet of Things: Threats and Challenges", Security and Communication Networks - Special Issue on Security in a Completely Interconnected World , 2013.

Authors' Addresses

Oscar Garcia-Morchon
Philips IP&S
High Tech Campus 5
Eindhoven, 5656 AA
The Netherlands

Email: oscar.garcia-morchon@philips.com

Sandeep S. Kumar
Philips Research
High Tech Campus
Eindhoven, 5656 AA
The Netherlands

Email: sandeep.kumar@philips.com

Mohit Sethi
Ericsson
Hirsalantie 11
Jorvas, 02420
Finland

Email: mohit@piuha.net