Aspects of Security Update Handling for IoT-devices

Geir M. Køien

Institute of ICT
University of Agder, Norway
Email: geir.koien@uia.no

***Abstract*—There is a fast-growing number of quite capable Internet-of-Things (IoT) devices out there. These devices are generally unattended, often exposed and frequently vulnerable. The current practice of deploying, and then leaving the devices unattended and unmanaged is not future proof. There is an urgent need for well-defined security update management procedures for these devices. Sufficient, sensible and secure default settings, as well as built-in privacy must be included. This paper presents a brief overview of the IoT threat landscape, argues for the necessity of security update provisioning for the IoT devices. As such, it is a call for action. Finally, an outline of a privacy-aware security update provisioning model is given. We have included incident management as well in the outline, but is only very rudimentary sketch of what one would need to provide. Suffice to say that there may be a need for these capabilities too, but it can probably only be justified for relatively capable devices.**

Keywords–Security update; Internet-of-Things; Incident reporting; Security maintenance; Privacy; Security management.

I. Introduction

A. Background and Motivation

This paper is based on the paper “Security Update and Incident Handling for IoT-devices; A Privacy-Aware Approach” [1], presented at SecurWare 2016.

It was noted that there is a growing number of relatively capable devices being designed and deployed. We only concern our selves with this class of devices in this paper. These devices, although quite simple, tend to have sufficient hardware support to be able to provide cryptographic functionality. It is thus feasible to design security schemes for these devices.

A central argument of the above paper was that IoT devices should be properly managed. It was postulated that the majority of the IoT device owners will be unable to adequately manage the devices, and furthermore they would generally be ill-equipped to understand and respond to security and privacy requirements. To solve these problems, IoT devices will need to have fully-automated security update capabilities. No user intervention should be required, although one must permit knowledgable users to configure the mechanisms. The security maxim should be “Security-by-Default”, where sensible security defaults are applied and enabled. Of course, privacy must also be catered for, and one may here look to the “Privacy-by-Design” initiative for high-level guidelines [2].

Since the original paper was published in July 2016, we have witnessed a number of high-publicity Internet infrastructure attacks facilitated by IoT devices with poor or non-existent security. These include, amongst others, large scale Distributed Denial-of-Service (DDoS) attacks using web cameras. With a proper security update solution in place, these cameras would have been substantially less vulnerable, and the DDoS attack by the Mirai-based malware would likely had been a lot less effective or maybe even fully prevented. We shall provide an update on some real-world attacks on unattended and generally unprotected IoT devices in Subsection II-D.

We have further updated the original paper on a concrete and practical firmware (FW) update schemes already in place. This scheme will serve as an example of the basic firmware update capability that is often provided with uncommissioned devices. Generally, it seems that the basic FW update functionalities may be reasonably complete by themselves, by that the trust assumptions are fairly naive. Furthermore, the schemes are often quite limited in scope and cannot provide anything other than a basic rudimentary update functionality. That is, there is hardly an overall solution in place, which provides credible security, roll-back, etc.

We must stress that to provide basic capabilities is not enough. The solution must be automated, completely transparent to the user, and it must provide credible security and privacy. Of course, the security update scheme must also be trustworthy and honest with respect to agreed capabilities and attributes. There has recently been reports of abuse of such schemes [3]. The scheme in [3] was not a security update scheme, but a fully automated firmware-based App downloader. It was also covert, and it did carry out software installation and updating without any user interaction. It may best be described as a persistent App installation scheme, reinstalling and updating unwanted Apps irrespective of user actions.

There needs to be a level of assurance and some measure of enforcement in place, and while a technologically basis must be provided, one likely also need support from jurisdictional and regulatory authorities. That is, there must a) exists pressures to provide honest and effective security update services and b) there must exists authorities which can react to protect end-users when update functionality has been used in subversive ways. We note that legal and regulatory control is slow acting and that they only seem to react after-the-fact.

We note that a properly implemented security update scheme will look a lot like a so-called “command & control” structure that is typically employed by botnets. And, clearly also quite similar to the scheme in [3]. However, the control servers for a security update scheme should be fully visible and official, so traffic to/from a security update server would not be confused with botnet control plane traffic (which may also be obfuscated to hinder intrusion detection systems (IDS) from noticing it).

B. Outline of our Proposed Security Update Model

The security update management and a minimal security incident and anomaly reporting service presented in this paper is not intended as a realistic model or proposal. The aim of proposal is rather to identify and highlight aspects of a possible solution, and thus to identify and illustrate requirements.

An important aspect of the model is to demonstrate technical feasibility. This is in line with the article itself, which aim to demonstrate the urgent need for security update services. The suggested architecture model features three information planes:

- User Services Plane (USP)
- User Management Plane (UMP)
- Security Management Plane (SMP)

The services will be realized by a two-tier architecture, separating global and local components, with clear division of authority and assumed trust between them.

The USP and UMP service planes may have cloud-based components, but whatever the case, these planes will have “local” termination with respect to the IoT device. The SMP service will be centralized and “global” in scope.

Privacy is a required property, and our design aim to adhere to the Privacy-by-Design (PbD) [2] tenets. We have therefore taken steps to make the model privacy-aware and privacy respecting, by introducing separation of duties and being particular at what kind of trust is placed in which architectural component/layer.

C. Related Work and Relevant Standards

The field is not yet settled, and the number of papers and proposed standards, of all types, is large and growing. We expect security and privacy to become even more important for IoT in the future. Our paper highlight the needs for secure management, and provide pointers as to how one could design such system.

1) Related Work: A few examples.

The survey paper “Security, privacy and trust in Internet of Things: The road ahead” [4] contains a broad overview over the challenges to IoT security. It emphasises that the IoT vision is characterized by heterogeneity, in terms of technologies, usages and application domains. It is also a fast phased and dynamic environment. Traditional security measures still play a large role, but the paper highlights that these are not always complete, sufficient or even appropriate. The authors also point out that scalability and flexibility is essential in this domain.

Another paper which also highlights open issues more than solutions is found in [5]. Also, the authors discusses these and related issues, like vulnerability, threats, intruders and attacks, in [6]. Both papers take a relatively high-level perspective. Other relevant works include [7]–[11].

In [12], the authors claim that “And as IoT contains three layers: perception layer, transportation layer and application layer, this paper will analyze the security problems of each layer separately and try to find new problems and solutions.”. In the end, the authors conclude that IoT devices are more exposed and less capable than other network elements, and that therefore the challenges are both different and more urgent. Trust related to IoT devices, both in software and hardware, is discussed in [13].

2) Relevant Standards: There is no shortage of formal standards and industrial standards concerning IoT and security for IoT. The following is an incomplete selected set of standards. There is a bias in the selection towards wireless and cellular communications standards. We feel this is well justified given that very large proportion of the IoT devices will have WLAN and/or cellular capabilities built-in. Others will probably have Bluetooth (Low Energy) or some similar short-range access technology that in turn enables access to the internet.

– 3GPP TS 33.401: 4G Security Architecture

This standard is about the 3GPP 4G security architecture and it encompasses security for the eNodeB (eNB) base (tranceiver) stations (chapter 5.3 in [14]). In a 4G network, to achieve sufficient spatial ($[bit/s]/m^2$) capacity, one needs a densely distributed network of eNB’s. There will therefore be a large number of eNB’s, and the scenario may be somewhat reminiscent of a managed IoT network. Security for updating and managing the highly distritbuted base stations may be different from many IoT scenarios, but we believe there are many similarities and lessons to be learned here.

– 3GPP TS 33.310: Authentication Framework

This standard [14] specifies, amongst others, roll-out of digital certificates to the 3GPP eNB base stations, using the Certificate Management Protocol (CMP) [15]. This part is highly relevant for IoT devices too, since many of them will indeed be capable of handling asymmetric crypto and digital certificates. Indeed, even the humble SIM card (smart card) is able to do so, and we therefore postulate that this capacity is fully feasible for any IoT device that needs to handle security sensitive data and/or privacy sensitive data. Moore’s law also implies that this capacity will only be cheaper over time, and so we fully expect that such capabilities will be commonplace.

– 3GPP TS 33.187: Machine-Type Communications

This standard [16] encompasses security for the so-called Machine-Type Communications (MTC). The standard defines how to allow IoT and machine-to-machine (m2m) devices be connected to a Service Capability Exposure Function (SCEF). Specifically, TS 33.187 requires “integrity protection, replay protection, confidentiality protection and privacy protection for communication between the SCEF and 3GPP Network Entity shall be supported” (Chapter 4.1 in [16]). These aspects are important for all IoT devices and this standard may serve as design input for non-3GPP cases too.

– GSMA CLP.11: IoT Security Guidelines Overview

This document [17] by the GSM Association is a non-binding guidelines document, and is as such not a normative standards document. It may still be quite influential since the GSM Association does have great reach within the community of cellular operators and vendors. The document identifies a set of grand challenges for IoT, and then proceeds to propose possible solutions. The challenges listed are:

A) Availability
B) Identity
C) Privacy
D) Security

Provisioning of scalable and flexible identifier structures is at the heart of the problem. Similarly, availability and security normally presupposes that the entities (the IoT devices) can be identified. Privacy then adds to this, but presupposing strong security [2] *and* requiring that the long-term identifiers are never exposed in clear (amongst others).

The document pays considerable attention to life-cycle aspects issues. The document also includes a chapter on risk assessment, an aspect which is all too often neglected in standards documents. Would-be IoT system designers are well advised to take this document into consideration. The document seems inspired by the "assumptions must be stated" idea, in a similar vein to the "Prudent Engineering Practice for Cryptographic Protocols" [18] paper. We strongly approve of the need for being explicit about assumptions and conditions.

– NIST: Cyber-Physical Systems (CPS) Framework

The NIST "Framework for Cyber-Physical Systems" document is an ambitious document which is expected to have considerable influence over future products [19]. The CSP Framework is largely oriented around the notion of systems-of-systems.

We also note that NIST has initiated work on "IoT-Enabled Smart City Framework" (abridged to "IES-City Framwork"). The framework is developed by a consortium, and started in earnest March 2016. Currently, only a white paper has been released by the working group [20].

3) Emerging Standards: International Mobile Telecommunications (IMT) is a framework for international mobile systems. It is mainly oriented towards defining capabilities, and have previously been defining framework for 3G (IMT 2000) and 4G (IMT-Advanced) mobil systems. The coming standards for 5G mobile systems, based upon the International Telecommunication Union (ITU) so-called "IMT for 2020 and beyond" vision, will have substantial support for "machine type communications (MTC)" [21]. The 3GPP, which is a consortium that includes standards development bodies, telecom operators and vendors, develops the concrete technical specifications based on the IMT vision. The 3GPP has stated that the basic technical standards for the IMT-2020 vision should be ready during 2020, and that some of the more advanced features are scheduled for 2021. Products, 5G compliant nodes/components and devices, will start arriving shortly after this. Experimental- and pilot deployment of parts of the 5G architecture already takes place.

Figure 1 depicts the 5G service triangle, where two of the three sides will have a strong focus on MTC services:

- **Enhanced Mobile Broadband:** Mainly focusing on bandwidth and to some extent user mobility
- **Ultra-reliable and Low Latency Communications:** This axis also encompasses the so-called "Critical MTC (cMTC)" type of communications. Strong security and hard requirements on bit error probabilities are part of this vision, and also fog computing (due to stringent round-loop latency requirements).
- **Massive Machine Type Communications (mMTC):** Low system/device overhead is main priority (extremely low power, small and infrequent payloads, upto 10^6 devices per km^2)

It is early days for IMT 2020 and 5G, but we expect important standards to emerge from for instance the 3GPP work on 5G, and some of these will no doubt have an impact on future IoT security.

D. Paper Layout

In Section II, we provide a high-level problem description. This includes the main aspects and high-level requirements. In particular, we provide a basic outline of the threats and real-world attacks that a IoT security scheme will have to face.

In Section III, we continue our investigation with a focus on underlying assumptions and premises concerning the devices and the detailed security service needs. This includes details concerning device capabilities, concerning firm ware updating and concerning device identifiers and location/identity privacy concerns.

In Section IV, we provide an outline of the proposed security management plane model. Here we outline the logical planes, network components and interfaces.

In Section V, we discuss the achievements and in Section VI we round off with a Summary and Conclusion.

II. High-Level Problem Description

A. Security for IoT Truisms

In the article "Click Here to Kill Everyone" [22], the author postulates that the IoT may be seen as a world-size robot and that it is about time to get it under control. The article is a bit alarmist, but maybe rightly so.

A cental point to the article is that there is an arms race between information assurance and the people who want to exploit the IoT devices for their own illicit goals. Based on these observations, the author outlines a set of truisms. Awareness of these truisms, which may or may not be tautological to the various IoT actors, will help us better protect the IoT devices and the associated infrastructures.

Schneier's IoT security truisms:

1) On the internet, attack is easier than defense.
2) Most software is poorly written and insecure.
3) Connecting everything to each other via the internet will expose new vulnerabilities.
4) Everybody has to stop the best attackers in the world.
5) Laws inhibit security research.

One may or may not agree with this set of truisms, or one may find it inconsistent, overlapping or incomplete, but the obvious lesson here is that we sorely need professional security management for IoT devices and IoT infrastructure.

B. User Interaction, Security Fatigue and Informed Consent

As a general rule, we believe that it is unrealistic to expect the end-users to configure or carry out much in terms of security setup of IoT devices. Likewise, we believe that it is equally unrealistic to expect the end-users to act on information pertaining intrusion attempts and similar. Partially, this can be attributed the phenomenon of "security fatigue" [23], but it can also be attributed to the fact that, to most ordinary end-users, information concerning security configuration, setup or intrusion alerts, simply must be considered "non-actionable". That is, there is no realistic way that the end-user would know what he or she should do. As such, information, warnings

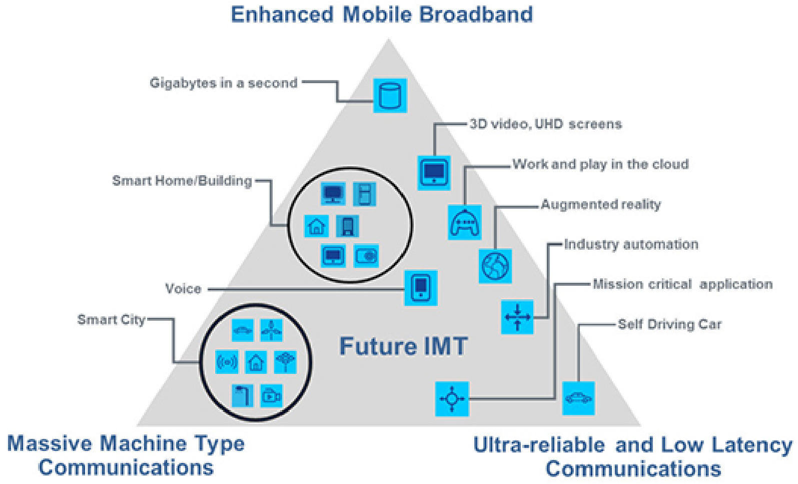

Enhanced Mobile Broadband
Gigabytes in a second
3D video, UHD screens
Work and play in the cloud
Smart Home/Building
Augmented reality
Industry automation
Voice
Mission critical application
Smart City
Self Driving Car
Future IMT
Massive Machine Type Communications
Ultra-reliable and Low Latency Communications

Figure 1. Usage secenarios of IMT for 2020 and beyond (Source: Fig.2 in ITU-R M.2083).

and alerts directed towards the end-user, that he or she cannot realistically be expected to know how to deal with, will only contribute towards "security fatigue". This would be analogous to the concept of non-actionable news, in which the meaning of the provided news items degenerates into mere entertainment [24]. Security related non-actionable information would have no entertainment value, but would contribute to cause stress and the before mentioned "security fatigue".

The problem with non-actionable information is also somewhat reminiscent of the problems with "informed consent". There are many papers highlighting these problems [25], [26], and one main objection is that one cannot easily expect anyone but experts to be truly informed.

We therefore conclude that while IoT devices should be managed, we cannot expect end-users to be able to do this except for possibly assisting a management system with very basic actions and decisions ("reset device","turn off device"). To ask users for permission to carry out various actions, the "informed consent" part, is likewise not very useful. It may serve a legal need, but this is pretence and has for the most part little to do with true informed consent.

C. Threat Landscape

The "European Union Agency for Network and Information Security" (ENISA) annually publishes so-called "ENISA Threat Landscape" (ETL) reports, the most recent being the 2015 report [27]. They also publish topic-orient threat landscape reports, but there is no report dedicated to IoT.

In chapter 3.2 Malware in ETL-2015 [27] it is noted that:

> "Rather than complexity, cyber-criminals are focussing on efficiency. In the reporting period we have seen the revival of infection techniques employed almost 20 years ago...."

We believe that this opportunistic cyber attack strategy is quite effective towards IoT devices, since they generally seems to have poorly designed and poorly implemented security functions.

We may ourselves briefly outline the basics of a threat landscape. The basic premises for assessing the threat landscape consists minimally of the following parameters:

- Asset identification and attributed value
- Asset exposure (per design)
- Attack surface and Vulnerability exposure
- Baseline security features
- Detection and Response capabilities
- Threat Agents (Intruder/Attacker)
- Attack Vectors
- Manifest Threats/Actual attacks

1) Asset identification and attributed value: What is it that has value? The physical device may have some value, but it is often the case that the data on the device has more value than the device itself. Understanding where the value actually is, is of course paramount.

2) Asset Exposure: For IoT the exposure or "visibility" is both through local physical exposure and through global connectivity exposure by means of the internet access. The local exposure, severe as it may be, does not scale and as such is of lesser importance. The global connectivity exposure is through the IP interface, and commonly though some sort of web server on the IoT device.

3) Attack surface and Vulnerability exposure: The attack surface is generally the whole of the exposed part of the asset. For our case, we define this to be the IP address(es) and the port range visible on the internet. The vulnerabilities would be associated with flaws or weaknesses in the information handling over the available attack surface. Exploitation of vulnerabilities is generally not straight forward, and it is not obvious that one can create attack vectors from a set of vulnerabilities. Or indeed, that the vulnerabilities are known to a threat agent.

4) Baseline Security features: The IoT device may or may not have some built-in security, but it is common to at least have some sort of password based scheme in place. The security in place will effectively mitigate vulnerabilities and remove or mitigate attack vectors.

Ideally, the configuration of the device would also include proper security hardening, and removal of all unneeded functionality and closing down all unneeded ports. This would reduce the attack surface and invariably also reduce the vulnerabilities, leaving less possible attack vectors available.

Advanced persistent threats (APT) is of course also a concern, but realistically these types are much less common and they are also far more difficult to protect against.

We therefore postulate that the baseline security ought to be able to fend off most of the trivial attacks. If the security measures are cost-effective, then certainly the baseline security should do more, but we cannot realistically require a simple IoT device to be able to withstand APT attcks.

5) Detection and Response capabilities: Low-cost IoT devices seldom have much in terms of detection and response capabilities. This is a problem, and it makes it substantially harder to recover from an intrusion event. This situation can actually be improved upon, and even low cost devices could have basic detection and response mechanisms in place. We return to this topic later in the paper (Section III).

6) Threat Agents (Intruder/Attacker): In this paper we will mostly consider relatively opportunistic threat agents. As was mentioned in the ENISA ETL-2015 quote, cyber criminals are more concerned with efficiency than demonstrating technical competency. That is, they are more concerned with goals than methods. It is therefore no surprise then that attacks as simple targeting devices with default administrator accounts with default passwords are popular. Script kiddies would probably also mostly use quite simple methods, or whatever methods easily available to them.

APT intruders are obviously also possible, but to protect against these are not part of the scope of this paper. At best, one can hope to make attacks costlier to these types of intruders, and thereby prevent or mtigate scalability of the attacks. This is important, since form a system perspective, to prevent attack scalability is an important goal.

7) Attack Vectors: Attack vectors are simply possible recipes to carry out a successful attack on a system, utilizing whatever exposed vulnerabilities there are. We note that what constitutes "success" is defined by the threat agent.

8) Manifest Threats/Actual attacks: Classification wise, this is actual attacks that has succeeded, using one or more of the available attack vectors. Success is here relative to the intruder goals, and these are detrimental to the security and privacy goals. Note that the intruder goals may not be aligned with what the end-user perceives to be the most valuable aspect of the IoT-device/service.

D. Real-World Experiences with Unprotected IoT Devices

During 2016 we have witness a new trend, in which cyber criminals systematically search out vulnerable IoT devices. The devices are attacked *en masse* and infected with botnet malware. A couple of rather high-profile DDoS attacks were conducted with the Mirai botnet malware.

In one instance, the web site of Brian Krebs, known as **KrebsOnSecurity**, were attacked [28]. By itself, an attack on a single host would be inconsequential and of little general interest, but in this case the attack was on an unprecedented scale, and caused internet giant Akamai to terminate the pro-bono hosting contract with Brian Krebs. They simply could not afford to stand up to the record breaking torrent of 620 Gigabits of traffic per second. Brian Krebs himself is a security researcher and blogger who does in-depth research and analysis of cybercrime worldwide. His reporting on DDoS attacks and the perpetrators apparently made him the target of the DDoS attack. The KrebsOnSecurity site is now hosted behind Google's **Project Shield**, which according to Google is "...is a free service that uses Google technology to protect news sites and free expression from DDoS attacks on the web" (https://projectshield.withgoogle.com/public/). The particular attack on KrebsOnSecurity seems to have been conducted by compromised routers, security cameras, printers and digital video recorder (DVRs). Default account names and passwords seems to be the common denominator for the infection process.

The Mirai source code was published subsequent to the attack on KrebsOnSecurity, which ironically makes it "open source" code [28]. Since then, Mirai has been used in other attacks, by other botnets. There was also a large scale attack on the French hosting firm OHV, and there was an attack on the company Dyn, who provides Domain Name System (DNS) services. The Dyn attack effectively prevent name resolution and thereby reachability for services such as Twitter, SoundCloud, Spotify and Reddit amongst others [29], [30].

The infection stage of Mirai have evolved after it was made public, and by now there are many variants of Mirai. The evolved versions are exploiting different vulnerabilities, and at least on strain seems to be specializing on infecting routers [28]. Mirai, of course, are just one type of botnet which attacks IoT devices. At this years DEF CON there was considerable attention on IoT security, and there results were abysmal. During DEF CON 47 new vulnerabilities were found in a total of 23 different devices [31]. One example includes solar panel. Several security issues were found, including a hard-coded password, a command injection flaw, an open access point connection and a lack of network segmentation [32].

E. Device Capabilities

Many of the devices, if power is not too much of a constraint, will be enjoying 32-bit processing, relatively large amounts of memory and even more flash memory. A typical mid-level IoT platform these days would be based on the ARM Cortex M family of processors. Here we have the relatively powerful ARM M4 processor (w/floating point and DSP functionality), being both very affordable and surprisingly power efficient [33], [34]. These devices typically provide 32-256KB SRAM memory and up to 1GB flash memory. We assume a device of roughly this capability in our design. However, the flexibility that comes with updatable software may also turn out to be an Achilles heal unless properly managed.

F. Lightweight, Minimality and Modularity

The core IoT architecture should be lightweight, including the base protocols. Correctness and efficiency is likely to benefit from this. Basic security and privacy functionality must be included in the core architecture.

Extensibility and additional features will be needed, and this must be designed to be modular. Restraint in adding features is necessary, but is clear that any successful architecture will over time grow more complex and encompass new areas [35]. We advocate a design reminiscent of the microkernel approach to operating systems design [36], in which only a minimal set of functional are at the core, running in supervisor mode, and where other component may be added and where strict rules concerning use of well-defined interfaces and protocols are adhered to. This will, amongst others, facilitate security hardening and it will enable the systems to be deployed on less capable devices.

G. Connectivity and Exposure

Commonly the devices will have bluetooth low energy connectivity, WLAN connectivity or even fixed LAN or cellular access. That is, they are reachable over the internet. This also exposes the devices to a whole range of threats, and whenever a device, or a class of devices, gains popularity they are prone to become a target. It is therefore prudent to assume that our IoT devices will, sooner-or-later, become targets.

H. Scalability

Needless to say, any solution that must be able to cope with a large, and fast growing number of devices, must be scalable. That is, the cost model for adding devices/users must be linear and with a low constant factor. The upper limit on the number of devices must be very high as to not prohibit future growth. The IMT-2020 vision for mMTC devices highlight this, with a requirement to serve in the order of a million devices per km^2. This calls for a redesign of the current access signalling schemes and for a new way of handling identifiers and access security. To combine solid security and credible identity/location privacy at the same time is not trivial.

I. Explicitness

As a rule, all requirements, including the security and privacy requirements must be explicit. Also, all conditions and premises must be made explicit. Explicitness is also a main lesson from [18] (being essential to Principles 1, 2, 4, 6, 10 and 11 in that paper).

J. Security and Privacy Requirements

Due to the exposure, the devices will need security protection, security supervision and security updating to remove, reduce and mitigate the risks. The devices will need basic capabilities regarding device integrity assurance, and for handling entity authentication, data confidentiality and data integrity.

It is quite likely that the devices will capture, store and transmit privacy sensitive data. Since there is a considerable chance that this may be so, it is prudent practice to take this into consideration. We therefore require that a PbD regime should be adhered to [2]. As noted in [37], [38], PbD does not come about all by itself, and considered and careful design, implementation and maintenance is required to create credible privacy solutions.

When it comes to communications security there are several options, depending on needs and what the devices actually communicates. We have typically the following possibilities:

L2 Link layer protection
L3 IP layer protection
L4 Transport layer protection
– No device support

The link layer protection support is often supported directly by the link layer hardware, whether it be Bluetooth, Zigbee, or some flavour of WLAN. Adequate configuration is still an issue, but the most up-to-date support found is often adequate and sufficient. There are notable exceptions though, and some chip sets do not support security at all.

There is generally very few devices which support IPsec directly. The IPsec code base is relatively large and this makes IPsec less well suited for many IoT devices.

There are transport layer solutions available, supporting https connection. This is quite reasonable since many IoT devices do provide a web based interface. Use of https is also on the increase, and it seems well justified to support https. Https support is also greatly facilitated by the efforts of the "Let's Encrypt" initiative, which is a free public Certificate Authority (CA) service [39].

K. Cryptographic Requirements

To be able to offer strong security and credible privacy, it is essential that the IoT device be able to support strong cryptographic algorithms and protocols. Additionally, there must be support for a secure execution environment and secure storage (more on this later). The basic requirements today is for "128-bit" security or better, and for "strong" algorithms. What is considered "strong" is a moving target, but as of February 2017 we have for instance that the commonly used SHA-1 algorithm has actually been broken [40]. Of course, there is SHA-256 and there is SHA-3 for hash functions, and there is the AES algorithm for confidentiality (with various well-defined mode-of-operation options available).

Quantum machines, which may become a practical reality within the next 10 years, will be uniquely able to break existing asymmetric cryptographic primitives. It is noted that standard cryptographic hash functions and symmetric crypto primitives will will be affected too. However, here it is believed that a doubling of key length (block length) will suffice to mitigate the effect of quantum computers. The National Institute of Standards and Technology (NIST) has published an overview of the problems associated with quantum computers and cryptography [41].

To the extent possible and practical, quantum-safe cryptography should be used.

L. Automation and Autonomy

We cannot expect that the end-users will provide security management for the devices. In fact, the end-user may increasingly be unaware of the presence of the IoT-devices. Effective security management of unattended and highly distributed devices will necessarily have to be automated and autonomous.

M. Challenges

As already mentioned, the GSM Association has recognized four main challenges created by IoT: *availability*, *identity*, *privacy* and *security* [17]. An autonomous security update and incident management system will need to address all these aspects, and provide at least a partial solution to the security aspect. We note that strong security is effectively a prerequisite for availability and privacy.

Trust and trustworthiness are essential elements and even prerequisites for widespread IoT adoption. Trust is a complex matter [13], but suffice to say that credible security management should instill confidence and thereby trust. Trustworthiness is hard to prove, but good security management should provide a measure of assurance.

N. Scope

The proposal made in this paper is an architectural proposal concerning security updating and incident and anomaly reporting. The proposal is, however, not a proposal for a fully fledged architecture, but rather for an architectural component. The proposal may therefore be compatible with other IoT architectures, but may of course also overlap with them or even be at odds with them.

In this respect, more is not going to be better, and defense-in-depth, which often means that there is benefit in multiple and possibly overlapping schemes, probably does not apply.

III. Assumptions and Premises

This paper makes a few assumptions about the IoT devices.

A. Internet Connectivity

We assume that the device is connected to the Internet. Locally, the connection may be wireless (Bluetooth, WLAN) or wired. It may also be a cellular connection. Preferably, there will be a hub/proxy device with firewall functionality etc., but this is not required.

B. Hardened OS

The OS is assumed to be hardened. Hardening is also assumed to be carried out when the OS is compiled and built with the program, as is often the case for embedded devices. Unnecessary protocols and services must removed or disabled, and only a minimal set of software be present. A local IPtables firewall may be deployed. There is a growing market for security hardened OS implementations [42].

C. Security Capabilities

The devices are assumed to have a trusted platform module (TPM), with basic crypto processing support and secure storage. Preferably, they adhere to standards such as ISO/IEC 11889-1:2015 [43]. A vendor issued device certificate is assumed to be available, or some similar identification that may be used for bootstrapping the CMPv2 protocol [15].

In late 2015, ARM released the ARMv8-M architecture, which is the new baseline Cortex-M architecture [44]. It introduces support for ARM's TrustZone TPM for the Cortex-M processors, and is as such an important step towards credible security for IoT devices. As of yet, there are no commercially available designs, but it is expected that there soon be a plethora of available processors targeted for the security sensitive IoT markets.

D. Power, Processing and Memory Capabilities

The device may have limited capabilities, but we shall assume that the device is not too restricted. That is, we assume it to be roughly at least as powerful as the lower end of the ARM Cortex M3/M4 processor families.

E. Secure Bootloading and Software/Firmware Attestation

A secure bootloader is necessary, and it will likely be using TPM functionality. All software, including firmware and patches, must be signed. All software packages shall have version numbers, and this includes firmware and patches. A TPM may facilitate attestation, but alternatives exists [45].

F. Firmware Over-the-Air

1) "Firmware Over-the-Air" (FOTA): is a firmware updating concept designed by Nordic Semiconductors. The FOTA scheme is targeted for Bluetooth Low Energy (BLE) enabled devices/chips, like the nRF52 device [33]. Here one have the so-called "Device Firmware Update" scheme. In particular, there is the "BLE Secure DFU Bootloader". The user guide, applying the "BLE Secure DFU Bootloader", is quite instructive [46]. The update FW package should be signed, and here one uses one of the available signature schemes. These are generally elliptic curve cryptography (ECC) oriented and using SHA-256. The ECC library used is the open source micro-ecc [47].

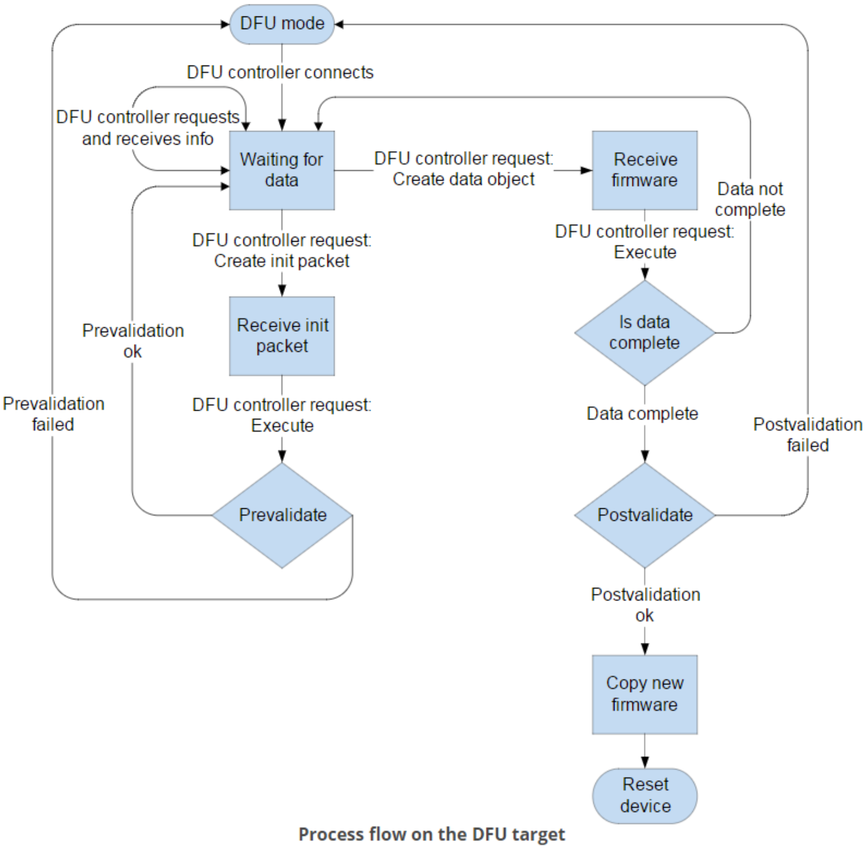
2) Secure Bootloader.: Nordic Semiconductors provides a secure bootloader scheme. The "BLE Secure DFU Bootloader" is not very easy to use or deploy, but it is still a useful tool for competent designers and developers. While Bluetooth connectivity is the main goal, the scheme also works over serial line protocols. It must also be mentioned that the ability to have roll-back and similar functionality is not quite there. There is the possibility to store multiple images, but the update functionality is still quite limited.

3) DFU bootloader.: The DFU bootloader supports updating the firmware of the device. This includes updating your application, the SoftDevice (which is the BLE handler) or even the bootloader. At startup, the DFU bootloader will check if a valid application already exists on the device. If there is no application present, the bootloader simply initiates the transfer of a FW image.

If there is a valid application present, the DFU bootloader will either start the application or go to DFU mode. There are several options, but only when in DFU mode will the bootloader actually install the new FW image. Having entered DFU mode, the DFU bootloader initializes the DFU transport module, which is responsible for receiving the new FW image at the chip. The downloaded image is validated and copied to the correct location in memory, before being activated. The device must be restarted to actually start executing the newly updated firmware. An outline of the process flow is presented in figure 2.

4) Omissions and Shortcomings.: The above described firmware updating scheme may be fairly typical, and we do not want to single out Nordic Semiconductors as being particularly bad. The secure bootloader scheme does provide basic update functionality and it has reasonable security with respect to the firmware image. That is, to the authenticity and data integrity of the image.

There is no data confidentiality provided, although that would not be too hard to facilitate. Given the lack of confidentiality, there can be no privacy protection for sensitive data. The scheme therefore cannot be used as-is to provide secure data backups, since it obviously allows information embedded in the image to be exposed.



Process flow on the DFU target

Figure 2. Process flow on the DFU target (Source: Nordic Semiconductors)

For a complete scheme we clearly also need more fine grained control with respect to permission and authorization. The secure bootloader update granularity is coarse, basically covering the image, although it is possible to differentiate somewhat (update the bootloader, update the application and update the Soft Device). There is for instance no way to read/write/delete application configuration data separately. Another aspect is that there is no framework for distinguishing between purely functional updates and security updates. While we strongly advocate automated security updating, this is not the case for functional updates. Tools and support for functional updates is important, but the end-user (or authorized manager) may have many good reasons for not wanting to implement new functionality.

The scheme is limited to serial line communications, which is also how it is implemented on top of the Bluetooth link. This limits the usefulness of the scheme for devices that ought to be able to communicate over the internet. Having said this, it must be acknowledged that the secure bootloader scheme limits the exposure of the scheme to to the local BLE range or serial line range.

G. Device Recovery

The device shall feature a secure loader, which facilitates a basic boot strap procedure that can securely rebuild the device software. We expect this to be part of the TPM functionality.

H. Device Identifier

The device must have a unique device identifier. This identifier is assumed to be used in the device certificate, but we shall otherwise be agnostic about the nature of the identifier. The device may also have, or use, higher-layer identifiers, but this is considered outside the scope of this contribution. An example would be a dropbox account identifier.

The device may also have network addresses and cellular identifiers. These *may* uniquely identify the device, but we do not in general consider these to be appropriate for identifying the device (observe the *explicitness* rule).

I. Identifiers and Privacy

A fundamental part of privacy is that there is sensitive data that is linked to a person. That is, usually we are concerned with linkability. If one can break the linkage between the person and the sensitive data, then leakage of the data would not necessarily be (privacy) critical.

We must assume that an intruder will be able to link plaintext device identifiers with the person(s) associated with the device. This capability is after all the core business for enterprises like Google. Consequently, we must assume that the intruder will be able to correlate unprotected data.

It is thus necessary to conceal the permanent device identifier such that no outsider will be able to associate the device identifer with the device or the user/owner. There are several ways to do this, including those described in [48], [49].

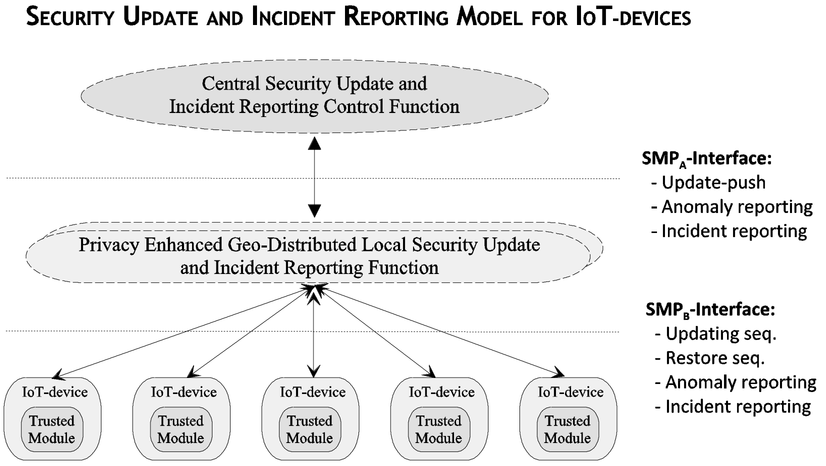


Figure 3. Outline of the Security Management Plane Model.

The functional split between the global and local services are very much reminiscent of split found in the cellular networks, where the local component necessarily must know the location and where the central component must necessarily know the permanent identity. Here, it has been shown that with proper setup one may achieve both location- and identity privacy [50]. In this paper, we shall ignore the specifics, but we do require that identifier and location privacy is part of the design.

IV. Outline of the Security Management Plane Model

Figure 3 depicts an outline of the Security Management Plane (SMP) model. We have already introduced the logical planes, but shall now take a closer look at how they are arranged. We shall primarily investigate the SMP plane and the associated services.

A. Trust Assumptions and Trust Relationships

We have the following principal entities in our model:

- **USER:** The user and/or owner of the IoT-device.
- **LOCAL:** The local SMP component.
- **GLOBAL** The global (centralized) SMP component.

We assume that the USER is an entity entitled to privacy protection according to the local laws. The GLOBAL entity is assumed to be operated by the IoT device manufacturer or some entity operating on behalf of the device manufacturer. It may also be operated by the software manufacturer. This would be similar to patch update services operated by Microsoft, Google and others. A standard, such as "Cortex Microcontroller Software Interface Standard" (CMSIS) [51], might also be extended in the future to cover support for patch management tools and facilities.

The LOCAL entity is assumed to be operated by a local entity, perhaps a local branch of the IoT manufacturer or some authority which is legally responsibly, warranties etc., for the IoT devices. It is required that the LOCAL and GLOBAL entities strictly observe the SMP model with regard to information exchange. We have observed that in the post-Snowden era, local authorities have increasingly required critical services to be hosted locally. We therefore have reason to believe that similar requirements may surface for IoT-devices too, or that such services are seen as commercially important to reassure the end-users (building confidence and perceived trustworthiness). We have the following trust assumptions:

- **USER vs. LOCAL**

  The USER trust LOCAL with respect to provided services. This is an asymmetric dependence trust.

- **LOCAL**

  The LOCAL entity must have security trust in the GLOBAL entity. The LOCAL entity shall not trust the GLOBAL entity with respect to USER privacy. The LOCAL entity cannot fully trust the USER. The LOCAL entity trust the incident- and anomaly reports, but do not place high significance in individual reports.

- **GLOBAL**

  The GLOBAL entity trust the LOCAL entity with respect to security, but not blindly so. The GLOBAL entity trust the incident- and anomaly reports, mediated by the LOCAL entity, but need not trust any single report and/or report from any single device.

B. The Logical Planes

1) The User Services Plane (USP): USP consists of the data associated with services provided by the IoT-device. The data forwarded here may end up at an App, at a local web service or at a cloud-hosted web service. We shall not be further concerned with the USP in this paper.

2) The User Management Plane (UMP): UMP consists of the device setup and configuration services provided by the IoT-device. The UMP is specifically about setting up the device end-user functionality. It does not cover basic security or privacy related setup or configuration. The data associated with UMP may end up at an App, at a local web service or at a cloud-hosted web service. The data *may* be privacy sensitive, and the design must reflect this. We shall not be further concerned with the UMP in this paper.

3) The Security Management Plane: The *security management plane* (SMP) is the crux of this paper. It consists of:

- Security setup and configuration
- Security update functionality
- Security incident and anomaly reporting, including local aggregation
- Secure restore functionality
- Identity- and Location Privacy handling

There will be a division of labor:

- Local SMP handling
- Centralized SMP handling

This will facilitate privacy and provide geo-distributed services. Localized processing may easier satisfy national regulatory requirements, while centralized analysis and handling of incidents will provide scalability and efficiency benefits.

C. The Network Components

The division or labor implies a LOCAL component and a centralized GLOBAL component. We observe that the local component will need to have provisions for geographical assurance. Implementation-wise, it will be a matter of policy if there is a need to comply with jurisdictional and regulatory requirements that dictate location of the local SMP handling.

1) The Central/Global SMP Component: The central security update and incident management control function will facilitate both security update production and distribution, and security incident and anomaly analysis.

This function does not need to know the device identifiers, nor does it need to know the associated IoT-device owner or user(s). It may need to know the software version status and any report on incidents and security anomalies associated with the devices. For the purpose of the incident analysis, we restrict this function to know the device class and the identity of the local SMP handling component. The true device identifier must never be divulged to the central SMP component.

2) The Local SMP Component: This function handles interactions with the IoT-devices within its geographical coverage area. We expect this area to coincide with regulatory or jurisdictional borders. The local SMP component may or may not be cloud-hosted, but in any case geo-location assurance must be possible.

The IoT-devices will communicate with the local SMP component. The local component will therefore know both the IP-address and the device identifier. The IP-address may be concealed if one uses Tor services [52], but the device identifier must be known to the local SMP component.

The local SMP component will communicate with the central SMP component, and it will receive protected security patches and software packages from the central SMP component. The local SMP component will aggregate and anonymize incident- and security anomaly reports from the IoT-devices before forwarding them to the central SMP component. The local SMP component may use temporary synthetic alias identifiers for a device, if there is a need for device references. This identifier must never be allowed to become an emergent identifier, and it must be fully de-correlated from the true device identifier. The de-correlation must be complete with respect to the full context given by the message exchange.

D. The SMP-Interfaces

1) The SMP_A-interface: This is a fully authenticated and security protected interface between the local SMP component and the central SMP component, as depicted in Figure 3.

2) The SMP_B-interface: This is a fully authenticated and security protected interface between the IoT-device and the local SMP component, as depicted in Figure 3.

3) Realization: The abstract SMP protocols should be agnostic about the underlying security transport protocol. Suffice to say, that strong security and credible privacy must be assured. The ENISA recommendations for cryptographic protocols, algorithms and key lengths provides good advice in this respect [53], [54]. ENISA is an EU agency, and the recommendation therefore carry some significance.

E. The SMP Services

1) Security Update – Local provisioning: One can have both push and pull mechanisms for security updates, but for IoT devices we do not generally recommend push solutions since it probably require more resources from the device. Push solutions may of course be appropriate for zero-day vulnerabilities, but scheduled pull solutions would likely suffice for patches that are less urgent and less critical. The scheduled pull frequency should reflect the security policy for the particular device class and according to usage, availability, etc. That is, IoT devices with sufficient processing power and no restrictions concerning power, may also use push services.

In either case, signed security updates will be received by the IoT device. All updates must be numbered, and the device will log the date/time and update number before implementing it. The local SMP shall not maintain logs about device status unless required to do so by the IoT device.

2) Security Update – Central provisioning: Whenever a security update patch is produced, the central SMP component will distribute the security update to the local SMP components. We recommend update frequencies to reflect the common vulnerability scoring system (CVSS) [55], although the CVSS system has been criticized for not properly reflect IoT devices [56]. The normal “serious vulnerability” score of 7 may therefore not properly reflect IoT concerns.

3) Incident- and Anomaly Reporting: Security incidents and anomalies are detected and reported by the TPM. This information is used by the SMP components to uncover large scale attacks and emerging attack trends. The ENISA publication [57] provides valuable guidance as to EU regulatory input on incident reporting.

4) Local Incident and Anomaly Reporting: This service will include software status, including patch levels etc. The device identifier is part of the security context, but should not be part of the incident/event report itself. A synthetic referential identifier may be provided by the local SMP.

It may, subject to authorization, be beneficial to store the incident history of the devices at the local SMP. This may allow the local SMP to detect if certain devices are specifically targeted. If so, one may speculate that the IoT device is an advanced persistent threat (APT) target. This in turn may trigger increased supervision and alarms.

5) Central Incident and Anomaly Reporting: The local SMP component will forward incident reports to the central SMP component. The local SMP component shall take steps to replace identifiers, if any, such that the central component never learns the true device identifier behind a reported incident. The local component *may* aggregate certain events and may delay reports to provide further de-correlations.

6) Device Attestation: The IoT device may request attestation services from the local SMP component. This service will need to be based on TPM functionality and permitting the local SMP component to survey the state of the IoT device. It may be part of a forensics service or a device recovery service.

7) Device Recovery: The IoT device may subscribe to recovery services at the local SMP component. As a minimum the local SMP should provide services to restore the device to a pristine condition, with all recent security update patches being implemented. The services may also account for security backup, with configuration data etc. being included in the restore procedure.

8) Device Backup: The local SMP component may provide a secure backup procedure, covering all or selected data elements. This procedure must permit to backup an entire device image and later restore the image. The device image must never leave the device in unprotected form. The device backup data should be encrypted and protected by the TPM, using unique device specific keys. Only the TPM should be able to restore the backup data.

9) Device Decommissioning: Life cycle considerations implies that one will need an explicit way of clearing all information on the target device. This will in effect clear all data and restore initial factory settings. This procedure must be resilient enough to withstand efforts from ordinary forensic tools to restore the information. The procedure may be triggered by a request via the local SMP component. The TPM should be responsible for carrying out the task.

V. Discussion

This paper describes an outline of an architectural component. Quite a few of the characteristics described below cannot be fully judged on the basis of the outline.

A. Lightweight, Minimality and Modularity

Our architectural component outline is both lightweight and relatively minimal. It is also modular, in the sense that it will build upon basic identifier structures and cryptographic capabilities, and delivers higher-level services.

B. Explicitness

This is related to requirements and conditions, including preconditions and postcondition. Essentially we have a “Mean what you say and say what you mean” situation. Use of formal methods may help verifying that captured requirements are adhered to, but these tools cannot in general help out with the “capturing” part. Explicitness must be enforced in any further development of the architectural component and in any implementation.

C. Scalability and Exposure

The division into a local-global split will facilitate scalability, as well as improving error resilience and thereby improving availability. Exposure is a necessary evil, but conscious design and appropriate use of cryptographic protocols can significantly reduce the unwanted effects of exposure.

D. Security and Privacy

The concrete security mechanisms is not specified in our proposal. Hence, more work is needed here for a concrete realization. However, there is no grand challenge here, only work that must be done precisely and consistently. Identity privacy and unlinkablity is mainly addressed through the local-global functional split. Data privacy is primarily by means of encryption. The requirements for the split is important, and schemes and measures that enforce the split must be encouraged. It would seem prudent to have this as a contractual requirement, and local regulatory requirements may also be an instrument in enforcing the functional split. Still, in the end, there must also be an economical incentive to manage and run both the local and the global infrastructure.

How credible is the privacy?

Clearly, it depends on the split between the local and global component being fully respected. There exists other solutions that would avoid this. These would be *privacy-preserving* and tend to be based on secure-multiparty computation and/or homomorphic cryptography. However, as argued in [58], strong irrevocable encryption may in the end provide less security and privacy. Governments are claimed to act a long the lines of "If we cannot break the crypto for a specific criminal on demand, we will preemptively break it for everybody." [58]. So, privacy must be balanced and possibly revoked, and this is achieved in our proposal.

E. Challenges: Availability, Identity, Privacy and Security

"Identity" is the only aspect that has not been addressed by our proposal. That is, we have identified this as a building block that our proposal depends upon.

F. Scope and Completeness

The scope is limited to a high-level model. Within the scope the proposal is reasonably complete, but there are many parts to be resolved, and the details have not yet been fully worked out.

G. Further Work

The model presented is an architectural component of a security architecture. Further work is needed to fit this component into a complete architecture. In particular, the concrete implementation of the security requirements should be aligned to the use in other areas. This is particularly relevant for identifiers and for basic services such as entity authentication, and integrity and confidentiality services.

Key agreement and key distribution must also be addressed and aligned to the overall security architecture. Preferably, one also wants to have a well-defined, effective and efficient security protocol to be the backbone of the services. As of today, one is often advised to use the Transport Layer Security (TLS) protocol [59] or the IPsec security protocols [60]. However, these are poor choices for IoT, and many version and implementations of TLS are also broken [61], [62].

That is, a dedicated, effective and efficient privacy-aware security protocol would probably be beneficial, provided that it would have wide-spread support. This archive this will be a difficult task, but following advise from [18], [53] and applying state of the art tools, it is also clearly doable on the technical level. Privacy, if it is to be credible, must be strongly aligned and be consistent over the full architecture to avoid leakage of sensitive data.

Smart metering or remote home monitoring would be examples of IoT systems that could benefit from the capabilities of the model. As such they would make good candidates for a pilot implementation to feature the model architecture.

VI. Summary and Conclusions

In this paper, we have identified the need for autonomous security update and incident/anomaly reporting for IoT-devices. In particular, we have addressed relatively capable IoT devices that ordinarily will be unattended devices, very much in line with a significant segment of the smart home devices.

This paper has provided a rough outline of a model in which IoT security update and incident handling is separated from normal user functionality, including user functionality setup and configuration. We believe that this is necessary since security management is becoming too complex to handle for end-users, and that the consequence of not managing security will be too severe. The current deploy-and-forget regime does not play out well for security functionality.

We have also provided a model in which there is a clear distinction between the centralized function and the local function. The main benefits of this arrangement is that one can more easily adhere to local regulatory requirements and one can provide identity- and location privacy solutions. This facilitates unlinkability, which is essential for credible privacy. It also enables scalability, which is ever so important for the IoT domain.

This paper represents an initial investigation of a new model for security update and incident handling for IoT devices. The model is not devised to be implemented as-is, but to serve as basis for discussions and further work.

References

[1] G. M. Køien, "Security Update and Incident Handling for IoT-devices; A Privacy-Aware Approach," in The Tenth International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2016), C. MerkleWestphall, H.-J. Hof, G. M. Køien, L. Králík, M. Hromada, and D. Lapkova, Eds. IARIA, 07 2016, pp. 309–315.

[2] A. Cavoukian, "Privacy by design; the 7 foundational principles," [retrieved: 06-2016] www.ipc.on.ca/images/Resources/7foundationalprinciples.pdf, 01 2011.

[3] D. Goodin, "Covert downloaders found preinstalled on dozens of low-cost Android phone models," Ars Technica, http://arstechnica.com/, 12 2016.

[4] S. Sicari, A. Rizzardi, L. A. Grieco, and A. Coen-Porisini, "Security, privacy and trust in internet of things: The road ahead," Computer Networks, vol. 76, 2015, pp. 146–164.

[5] M. Abomhara and G. M. Køien, "Security and privacy in the internet of things: Current status and open issues," in Privacy and Security in Mobile Systems (PRISMS), 2014 International Conference on. IEEE, 2014, pp. 1–8.

[6] ——, "Cyber security and the internet of things: Vulnerabilities, threats, intruders and attacks," Journal of Cyber Security, vol. 4, 2015, pp. 65–88.

[7] L. Patra and U. P. Rao, "Internet of thingsarchitecture, applications, security and other major challenges," in Computing for Sustainable Global Development (INDIACom), 2016 3rd International Conference on. IEEE, 2016, pp. 1201–1206.

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

[9] Z.-K. Zhang, M. C. Y. Cho, C.-W. Wang, C.-W. Hsu, C.-K. Chen, and S. Shieh, "Iot security: ongoing challenges and research opportunities," in Service-Oriented Computing and Applications (SOCA), 2014 IEEE 7th International Conference on. IEEE, 2014, pp. 230–234.

[10] M. M. Hossain, M. Fotouhi, and R. Hasan, "Towards an analysis of security issues, challenges, and open problems in the internet of things," in Services (SERVICES), 2015 IEEE World Congress on. IEEE, 2015, pp. 21–28.

[11] J. Granjal, E. Monteiro, and J. S. Silva, "Security for the internet of things: a survey of existing protocols and open research issues," IEEE Communications Surveys & Tutorials, vol. 17, no. 3, 2015, pp. 1294–1312.

[12] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, "Security of the internet of things: Perspectives and challenges," Wireless Networks, vol. 20, no. 8, 2014, pp. 2481–2501.

[13] G. M. Køien, "Reflections on trust in devices: an informal survey of human trust in an internet-of-things context," Wireless Personal Communications, vol. 61, no. 3, 2011, pp. 495–510.

[14] 3GPP TSG SA3, "3GPP System Architecture Evolution (SAE); Security architecture (Release 13)," 3GPP, TS 33.401, 03 2016.

[15] T. Kause and M. Peylo, "Internet X.509 Public Key Infrastructure – HTTP Transfer for the Certificate Management Protocol (CMP)," IETF, RFC 6712, 09 2012.

[16] 3GPP TSG SA3, "Security aspects of Machine-Type Communications (MTC) and other mobile data applications communications enhancements (Release 13)," 3GPP, TS 33.187, 01 2016.

[17] GSM Association, "IoT Security Guidelines Overview Document; CLP.11, Ver.1," [retrieved: 06-2016] www.gsma.com/connectedliving/wp-content/uploads/2016/02/CLP.11-v1.1.pdf, 02 2016.

[18] M. Abadi and R. Needham, "Prudent engineering practice for cryptographic protocols," IEEE Transactions on Software Engineering, vol. 22, no. 1, 1996, pp. 6–15.

[19] Cyber Physical Systems Public Working Group, "Framework for Cyber-Physical Systems," NIST, USA, Framework Release 1.0, 05 2016.

[20] IES-City consortium, "IoT-Enabled Smart City Framework," 02 2016.

[21] ITU-R, "IMT Vision - Framework and overall objectives of the future development of IMT for 2020 and beyond," ITU, Geneva, Switzerland, Recommendation M.2083-0, 09 2015.

[22] B. Schneier, "Click Here to Kill Everyone; With the Internet of Things, were building a world-size robot. How are we going to control it?" New York Magazine, 01 2017.

[23] B. Stanton, M. F. Theofanos, S. S. Prettyman, and S. Furman, "Security fatigue," IT Professional, vol. 18, no. 5, 2016, pp. 26–32.

[24] N. Postman, Amusing ourselves to death: Public discourse in the age of television, 1985.

[25] E. Sedenberg and A. L. Hoffmann, "Recovering the history of informed consent for data science and internet industry research ethics," 2016.

[26] T. Ploug and S. Holm, "Informed consent and routinisation," Journal of Medical Ethics, vol. 39, no. 4, 2013, pp. 214–218.

[27] L. Marinos, A. Belmonte, and E. Rekleitis, "Enisa threat landscape 2015," ENISA, Report ETL-2015, 1 2016.

[28] US-CERT, "Alert (TA16-288A): Heightened DDoS Threat Posed by Mirai and Other Botnets," https://www.us-cert.gov/ncas/alerts/TA16-288A, 10 2016, [retrieved 12-2016].

[29] C. Williams, "Today the web was broken by countless hacked devices your 60-second summary," The Register, http://www.theregister.co.uk/, 10 2016.

[30] B. Krebs, "DDoS on Dyn Impacts Twitter, Spotify, Reddit," KrebsOn-Security, https://krebsonsecurity.com/, 10 2016.

[31] L. Constantin, "Hackers found 47 new vulnerabilities in 23 IoT devices at DEF CON," CSO Online, http://www.csoonline.com/, 09 2016.

[32] F. Bret-Mounet, "All Your Solar Panels are belong to Me," DEF CON, https://media.defcon.org/, 08 2016.

[33] Nordic Semiconductor ASA, "nRF51822 Product Specification," Access: www.nordicsemi.com/eng/nordic/download_resource/20339/13/85365517, 2016.

[34] ARM Ltd., "Cortex-M4 Processor," [retrieved: 06-2016] www.arm.com/products/processors/cortex-m/cortex-m4-processor.php, 2016.

[35] G. M. Køien, "Reflections on evolving large-scale security architectures," International Journal on Advances in Security Volume 8, Number 1 & 2, 2015, 2015, pp. 60–78.

[36] A. S. Tanenbaum, "Lessons learned from 30 years of minix," Communications of the ACM, vol. 59, no. 3, 2016, pp. 70–78.

[37] S. Spiekermann, "The challenges of privacy by design," Communications of the ACM, vol. 55, no. 7, 2012, pp. 38–40.

[38] D. Le Métayer, "Privacy by design: a formal framework for the analysis of architectural choices," in Proceedings of the third ACM conference on Data and application security and privacy. ACM, 2013, pp. 95–104.

[39] Internet Security Research Group (ISRG), "Let's encrypt," Accessed March 2017: https://Letsencrypt.org, 03 2017.

[40] M. Stevens, E. Burzstein, P. Karpman, A. Albertini, and Y. Markov, "The first collision for full sha-1," Shattered IO, 02 2017.

[41] L. Chen et al., "Report on post-quantum cryptography," National Institute of Standards and Technology Internal Report, vol. 8105, 2016.

[42] Symantex, "Embedded security: Critical system protection," Access: www.symantec.com/content/en/us/enterprise/fact_sheets/b-sescsp-ds-21345379.pdf, 11 2015.

[43] ISO/IEC, "ISO/IEC 11889-1:2015," ISO, Geneva, Switzerland, Standard 11889-1:2015, 08 2015.

[44] ARM Connected Community., "Whitepaper - ARMv8-M Architecture Technical Overview," [retrieved: 06-2016] https://community.arm.com/docs/DOC-10896, 2015.

[45] F. Armknecht, A.-R. Sadeghi, S. Schulz, and C. Wachsmann, "A security framework for the analysis and design of software attestation," in Proceedings of the 2013 ACM SIGSAC Conference on Computer Communications Security, ser. CCS '13. New York, NY, USA: ACM, 2013, pp. 1–12.

[46] Nordic Semiconductors, "nrfutil User Guide v1.0," 09 2016.

[47] K. MacKay, "micro-ecc: ECDH and ECDSA for 8-bit, 32-bit, and 64-bit processors," 07 2016.

[48] G. M. Køien and V. A. Oleshchuk, Aspects of Personal Privacy in Communications-Problems, Technology and Solutions. River Publishers, 2013.

[49] G. M. Køien, "A privacy enhanced device access protocol for an iot context," Security and Communication Networks, vol. 9, no. 5, 03 2016, pp. 440–450.

[50] ——, "Privacy enhanced cellular access security," in Proceedings of the 4th ACM Workshop on Wireless Security, ser. WiSe '05. New York, NY, USA: ACM, 2005, pp. 57–66.

[51] ARM Ltd, "CMSIS MCU Software Standard 4.5," 2016.

[52] "The Tor Project," [retrieved: 06-2016] www.torproject.org, 2016.

[53] N. P. Smart, V. Rijmen, M. Stam, B. Warinschi, and G. Watson, "Study on cryptographic protocols," ENISA, Report TP-06-14-085-EN-N, 11 2014.

[54] N. P. Smart et al., "Algorithms, key size and parameters report 2014," ENISA, Report TP-05-14-084-EN-N, 11 2014.

[55] First, "Common vulnerability scoring system, v3," [retrieved: 06-2016] https://www.first.org/cvss, 06 2015.

[56] D. J. Klinedinst, "CVSS and the Internet of Things," SEI Insights, [retrieved: 06-2016] insights.sei.cmu.edu/cert/, 09 2015.

[57] M. Dekker and C. Karsberg, "Technical guidance on the incident reporting in article 13a (ver.2.1)," ENISA, Report, 10 2014.

[58] P.-H. Kamp, "More encryption means less privacy," Commuications of the ACM, vol. 59, no. 4, 04 2016, pp. 40–42.

[59] T. Dierks and E. Rescorla, "The Transport Layer Security (TLS) Protocol; Version 1.2," IETF, RFC 5246, 08 2008.

[60] S. Kent and K. Seo, "Security Architecture for the Internet Protocol," IETF, RFC 4301, 12 2005.

[61] H. Krawczyk, K. G. Paterson, and H. Wee, "On the security of the tls protocol: A systematic analysis," in Advances in Cryptology–CRYPTO 2013. Springer, 2013, pp. 429–448.

[62] C. Hlauschek, M. Gruber, F. Fankhauser, and C. Schanes, "Prying open pandora's box: Kci attacks against tls," in 9th USENIX Workshop on Offensive Technologies (WOOT 15), 2015, pp. 1–15.