HomeCA: Scalable Secure IoT Network Integration

Robert Müller[1] Corinna Schmitt[2] Daniel Kaiser[3] Marcel Waldvogel[4]

Abstract: Integrating Internet of Things (IoT) devices into an existing network is a nightmare. Minimalistic, unfriendly user interfaces, if any; badly chosen security methods, most notably the defaults; lack of long term security; and bugs or misconfigurations are plentiful. As a result, an increasing number of owners operate unsecure devices.
Our investigations into the root causes of the problems resulted in the development of *Home Certificate Authority* (HomeCA). HomeCA includes a comprehensive set of secure, vendor-independent interoperable practices based on existing protocols and open standards. HomeCA avoids most of the current pitfalls in network integration by design. Long-term protocol security, permission management, and secure usage combined with simplified device integration and secure key updates on ownership acquisition pave the way toward scalable, federated IoT security.

Keywords: Home Certificate Authority (HomeCA); Communications technology; Internet of Things (IoT); Network security; Wireless communication; Wireless Local Area Network (WLAN)

1 Introduction

The Internet of Things (IoT) consists of a myriad of network-enabled devices serving in various parts of our daily life (e.g. business, household, personal health and entertainment), which are connected via the Internet. IoT devices are vulnerable, not least owing to their use of wireless connectivity paired with limited defence resources, which brings an enormous diversity in possible attacks [BW15].
The number of incidents of lost control over IoT devices is on the rise [UC17]. They are then used to mount Distributed Denial of Service (DDoS) attacks [Wa16] or abuse the device itself, risking the owner's privacy and security. The owner will rarely be aware of these abuses, due to a combination of the devices' limited resources and minimal or non-existent user interfaces. Lack of long-term security updates, lack of vulnerability notifications to the users plus often complicated and impractical update procedures further challenge their use for any serious and safe usage.

[1] Universität Konstanz, Department of Computer and Information Science, Distributed Systems Laboratory, Universitätsstraße 10, 78457 Konstanz, Germany robert.mueller@uni-konstanz.de
[2] Universität der Bundeswehr München, Research Institute CODE, Werner-Heisenberg-Weg 39, 85577 Neubiberg, Germany corinna.schmitt@unibw.de
[3] University of Luxembourg, Security and Networking Research Group Netlab, Maison du Nombre 6, Avenue de la Fonte, 4364 Esch-sur-Alzette, Luxembourg daniel.kaiser@uni.lu
[4] Universität Konstanz, Department of Computer and Information Science, Distributed Systems Laboratory, Universitätsstraße 10, 78457 Konstanz, Germany marcel.waldvogel@uni-konstanz.de

doi:10.18420/inf2019_21

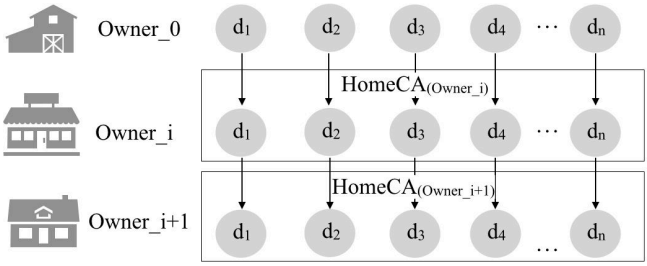

Fig. 1: HomeCA coverage on IoT devices life cycle

To add insult to injury, the humongous problems caused by insecure and sometimes insecureable devices (i.e. without interface/means to install security patches) are complemented by poor pairing protocols, which are insecure by design or implementation: Many application profiles of the ZigBee specification (low-power, low bandwidth, and low range) often used in smart homes allow, by definition, any malicious device to join the network and, thus, become trusted [Zi15]. Wi-Fi is not much better, recommending insecure defaults [VP16] and opening barn doors with Wi-Fi Protected Setup (WPS) [Wi14] which allows offline attacks [Bo14].

The manufacturers have been asked to provide more security by forcing the idea of privacy-by-design on them and applying data protection regulations [Wa16], though they are not always applied due to missing incentives and controls when not following the requests and rules. From our research on the field of wireless security on IoT devices we identified the following four core problems: (1) Vendors deliver their devices with simple instead of secure setup (savings in customer support requests). (2) Vendors do rarely provide security updates, provide no easy way of installing them, remove functionality unnecessarily together with updates [Ha16], or eventually go out of business. (3) Users do often leave passwords at the manufacturer's default settings, if they can change them at all. (4) Devices are often accessible from the Internet, even if it is not required for their functionality.

With our Home Certificate Authority (HomeCA) approach we determine new ways to pair home devices, to get rid of passwords, and to take advantage of new developments such as built-in security keys in micro controllers [Mi18]. At the same time, we are trying to remain vendor neutral and avoid unnecessary trust in the device keys, as manufacturing errors, database leaks, and low entropies have been known to cause problems in other published cases [Ma16].

HomeCA includes a lifecycle consisting of four phases: (1) Manufacturing, (2) Ownership Change, (3) Connection Establishment, and (4) Refresh. Figure 1 illustrates the transitions in the lifecycle of IoT devices from manufacturing, resale, use in a home network to the possibility of a reuse at another home, e.g., a friends home or during vacation.

With the process included in those phases, HomeCA supports secure pairing, and optional gatewaying can be provided, resulting in (1) removing insecure passwords and (2) reducing the dependency on updates, all in a vendor-independent layer with little overhead. HomeCA creates a protocol layer that (3) provides security, (4) does not require complex

and potentially insecure user interaction, and (5) can help reduce the trust required in the manufacturer.
HomeCA is a security protocol layer that covers the IoT devices within a trustworthy home network. It uses a machine-to-machine (M2M) authentication protocol, which allows authentication between devices and applications without user interaction except for an approval step to mitigate automated attacks, for the integration of new IoT devices into the network. It is designed to protect the devices from unauthorized manipulation and access using a Certification Authority (CA) within the home network. Threats that are denied through HomeCA and the used *methods* are: (1) Lacking or corrupted (security) updates *(verification)*, (2) Corruption of the manufacturer itself *(verification)*, (3) Attacks from the Internet aimed at eavesdropping communication *(encryption)*, and (4) Attacks from the Internet aimed at taking control of the IoT device *(certificates)*.
The paper is structured as follows: Section 2 focuses on related work in the area of securing inhomogeneous IoT networks. Section 3 introduces our HomeCA model leading to supported workflows in Section 4 and secure key update in Section 5. A prototype implementation is presented in Section 3.4, and Section 6 concludes the paper.

2 Related Work

Over the last decades many investigations took place to identify challenges in IoT [Su12], especially in the area of security of Public Key Infrastructure (PKI) [GM94], [Sv16] and in the area of key management [VP16], [Sc15].
However, it was shown that the IEEE 802.11 implementation Wi-Fi Protected Access 2 (WPA 2) group keys can be attacked based on the quality of the used random number generator [VP17], which often is embedded in IoT devices.
A good overview and a broad analysis of Authentication and Access Control used in the IoT can be found in [LXC12]. An abstract approach for IoT integration by a central signing authority server is followed in a Patent Application in [Sh18]. Research on secure and resources saving integration of constrained devices into the IoT is presented in [Kl15], with a certificateless signcryption scheme targeting at Wireless Sensor Networks (WSN) rather than Wireless Local Area Network (WLAN) networks. WPS [Wi14] and similar technologies do not provide the possibility to integrate a large number of devices. Focussing on smart homes, security challenges are studied in [BJD16], while in [Si15] a network centric approach is proposed, in which a central entity, remotely similar to HomeCA, is located outside the smart home network that provides services through a web interface to securely control smart home devices in the local network.

3 HomeCA Model

For the design and development of HomeCA we assume the following use case illustrating a life cycle of today's IoT devices: A set of IoT devices $\{d_n, n \in \mathbb{N}\}$ is created at a factory

and being sold at a retail store. Once they are bought, they are integrated and operated in a first home network. Eventually, they are moved to a second home, showing disassociation and re-association of the IoT device.

3.1 Security Model

IoT devices are target of attacks that try to access the sensitive data or try to gain control over the device. Devices taken over are used for illegal actions like Distributed Denial-of-Service (DDoS) attacks. This type of attempts, usually originated from the Internet, trying to access IoT devices within the private network, are detected by HomeCA through their suspicious or blacklisted source IP address and thus are dismissed. There is no unauthorized access to the IoT devices since traffic must pass the check of HomeCA and thus cannot reach the IoT device.
For privacy reasons it shall not be disclosed to the manufacturer, how many devices are within the personal/private network managed by HomeCA. This is achieved by signatures between the manufacturer, devices and HomeCA.
Attackers try to find and access IoT devices connected to the Internet. Methodology and tools to exploit known vulnerabilities of devices and protocols used in the IoT are described in [MM15]. Other information about the devices may be stolen from a manufacturer or service website that the IoT device is eventually connected to, such as a cloud service or a service provider API.
Next, an attacker tries to log in with default credentials to collect all devices that have not changed the factory-default username and password combination. Once successful, an attacker can configure the IoT device as he/she pleases. HomeCA protocol protects the access to IoT devices and therefore prevents this type of direct access to the IoT devices, except if they are legit.
Changes to the IoT device configuration requested by the owner are redirected to the HomeCA web-interface. As a consequence, multiple access attempts for IoT devices within the private network can be detected by the unusual high number within a definite time window. Allowed access attempts are additionally protected against maliciously configured bots by the use of i.e. "Completely Automated Public Turing test to tell Computers and Humans Apart"(CAPTCHA) [Ca17] or similar technologies capable of preventing bots from performing specific operations automatically. This is triggered by exceeding a variable threshold t on the number of devices being integrated at once.
Next, the HomeCA Lifecycle is introduced, which illustrates the required coupling between the HomeCA and an IoT device.

3.2 HomeCA Lifecycle

The HomeCA lifecycle of an IoT device consists of the following four steps, where the last one is optional (for details see Section 4):

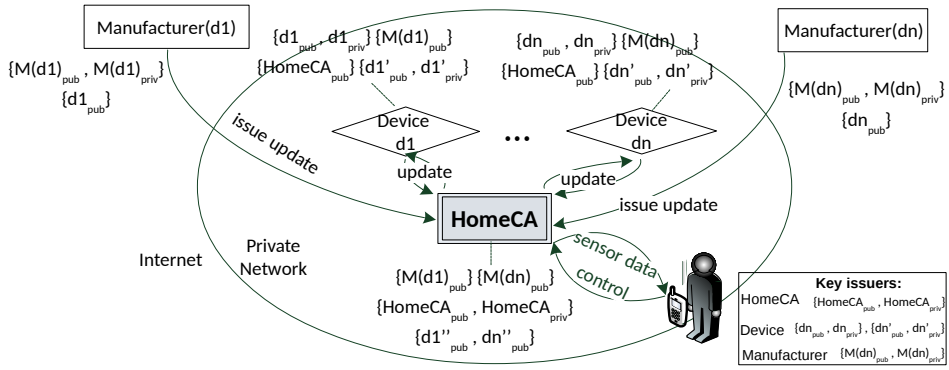

Fig. 2: HomeCA System Overview

1. **Manufacturing:** Initial key pair creation, public key delivery *(optional)*.
2. **Ownership change:** Release *(active or passive)*, HomeCA discovery, key verification *(optional)*, key update, certificate creation and revocation, rights management *(optional)*.
3. **Connection establishment:** Service discovery, Datagram Transport Layer Security (DTLS) connection setup.
4. **Refresh *(optional)*:** Liveness verification, certificate lifetime extension, rights management update.

Figure 2 shows the basic mechanisms of the protocol set-up for the secure communication of IoT devices with their two manufacturers and within a private/personal network.
Initially, keys for encrypted communication are exchanged between the manufacturers (M) $M_{1..n}$ and corresponding devices (d) $d_{1..n}$. Assuming an honest manufacturer for the time during transfer of posession), this offline exchange at production site is considered secure. Second, secure keys for bi-directional communication are exchanged between HomeCA and $d_{1..n}$, namely $d1''_{pub}$ .. dn''_{pub}. This happens within the private/home network and provides a trusted communication channel between the IoT device and HomeCA for access and control of the IoT devices $d_{1..n}$. No user-interaction for execution is required. Afterwards, the initially pre-shared keys are then shared by the devices (d) $d_{1..n}$ with HomeCA to verify an update issued by $M_{1..n}$ before it is applied by $d_{1..n}$.
In order to make the interface scalable for many devices we chose an interface that requires no human interaction except for security approvals when a large number of devices exceeding a variable threshold is handled. The device integration is based on a trust relationship between manufacturer, device and home network. The protocol execution steps for the handover of a device between multiple HomeCA are shown in Figure 3. Willing to be connected with another HomeCA, device d sends a request to the new HomeCA that is available within the network. To switch from the initial owner 0 to owners i and later $i + 1$, the device d asks the current HomeCA to be released. Afterwards d registers with the new HomeCA via a DTLS [RM12] handshake and thus is under new administration.

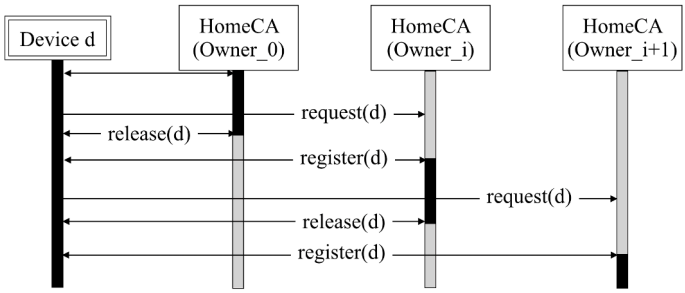

Fig. 3: HomeCA protocol integrating a new device

3.3 Security Use Cases

The use cases that require the secure infrastructure to update its trust relationship are:
Device d is brought into a private wireless network for the first time. HomeCA recognizes the new device and has to authenticate against d to prove it is authorized to manage d. Two ways exist to do the aforementioned: (1) Option one is chosen when the device is directly purchased at the manufacturer M. At purchase, a unique device ID of d that consists of a hash on identifiers e.g., Media Access Control (MAC) address, Bluetooth device identifier, and Personal Area Network Identifier (PANID) is transmitted from M to the buyer which imports them into HomeCA or submits an E-mail address to send the device IDs to. When the device d appears within the HomeCA network, they perform a handshake and can authenticate and establish a secure communication with d. (2) Option two is chosen when d is bought at a retail store. d and HomeCA meet in a private network considered secure when no interference (i.e. more than one HomeCA instance or other connection attempts) is being detected by d. HomeCA is actively asking for a new to be integrated device, when it knows i.e. from the application registry which sensor data to expect from the sensor of the IoT device d. Diffie-Hellman key exchange between d and HomeCA is used to obtain k to create new public and private keys.
Device d is sold /given away /defect: If HomeCA did not see its IoT devices for a period of time t, the certificates for that specific device are removed from HomeCA and must re-authenticate if it appears in the private network again at a later point in time. The initial time window is set to $t = 10$ days but shall be adopted based upon the dynamic of the actual user behaviour of the private network, i.e. the rate at which devices are introduced and removed.
Device software d has a security weakness or needs to be updated/patched: Only HomeCA can approve and sign a trustworthy update. Attackers without this certificate cannot create a valid update for the device. HomeCA monitors and keeps track of all updates on the IoT devices.
If the device d is compromised and behaves strangely, HomeCA removes the certificate for the device d automatically based on a revocation list. The revocation list is signed by HomeCA and provided for all associated devices as well for the scenario of devices

communicating directly with each other.
If the manufacturer M is compromised and no longer trustworthy, HomeCA is able to inform the IoT devices assigned with HomeCA about this situation.

3.4 Model Implementation and First Evaluation

In order to analyse and evaluate HomeCA, we chose to implement a prototype on physical devices. A set of three Raspberry Pi 3 Model B[5] is used to set up a model of one HomeCA and two IoT devices or optionally two HomeCA and one IoT device switching HomeCAs. Java is used as a platform-independtent language.
For network access and discovery purposes we employ a WPA2-Enterprise Radius Server (see Section 4.4). Also, the necessity of a database on the IoT device to maintain the certificates besides the connection information is identified and shall be used in the next release of HomeCA.

4 HomeCA Workflows

HomeCA workflows are used to create a trusted association between the device and the network and thus the other connected devices. Thereby, reasonable but minimal trust is expended on the manufacturer of the device and the HomeCA itself. To achieve this, as little sensitive information as necessary is processed by other devices. Key pairs are be solely managed by the device owning them, meaning private key material is not shared or transmitted unless absolutely necessary.

4.1 Manufacturing: Initial Key Pair Creation

Ideally, the IoT device itself is solely responsible to create the key pair and the private key is not accessible from the outside. This requires a good source of random bits, requiring the devices to be more complex. Key creation at manufacturing time comes with the following three options: Local generation, key feeding, and randomness feeding.
Local generation: This is the ideal, but requires a good internal source of reliable and confidential random bits, together with ample processing power. If the random number generator (RNG) does not meet these criteria, the key will be weak and can be guessed with little effort [Bo14].
Key feeding: Device does not possess a good enough RNG. Therefore, a key pair is created by an external device and fed to the IoT, e.g., during burn-in test. As a result, the external device owned by the manufacturer knows the private key. This key can then be obtained by

[5] Specification available at `https://www.raspberrypi.org/products/raspberry-pi-3-model-b/`

Tab. 1: IoT Device Protection Mechanisms and Access Rights

Service	Allowed function	Required protection	Gateway usage recommended
Heating	Report temperature	TLS 1.2	NO
Garage Door	Open/Close	SSLv3	YES
Key Generator	Online Authentication	TLS 1.3	NO
Entertainment	Volume control	none	YES
SmartHome	Steer Heating, Lights	SSLv2	YES
Coffee Maker	Preheat machine	SSLv2	YES
Fitness	Track steps	TLS 1.0	YES
Health	Blood Sugar monitor	TLS 1.1	YES
SmartWatch	Messages, Notifications	TLS 1.3	NO

third parties [SB15] and therefore should not be considered secret.

Randomness feeding: An alternative way to cope with a bad on-chip RNG is to use an external random source, e.g., provided again during burn-in test. The device then calculates its own key pair based on that data and keeps the private key secret. However, anyone knowing the key generation algorithm (often public) and the random bits fed, can recreate the private key and verify it with the public key, duplicating the problems of *key feeding* above.[6]

HomeCA uses local generation – in our prototype implementation with a hardware random number generator. Even if the key is generated on the device itself, the manufacturer may be able to extract the key using low-level device debug mechanisms such as JTAG [IE13].[7] Table 1 lists exemplary IoT devices we found. The devices are shown with provided functionality, protection mechanism and decision about whether the HomeCA Gateway shall be used based on the required protection. Gateway functionality (cf. Section 4.7) is proposed for some (legacy) devices with protection mechanism considered less secure to decrease the risk of attackers exploiting the communication protocol. This is explicitly used for services that might seem not security relevant, as e.g., with Smart Homes it would be disturbing for the user and fun for a hacker to turn lights and heating on/off.

4.2 Manufacturing/Resale: Public Key Delivery

After the key pair creation, optionally, the public key may be extracted from the device and associated with the device serial number. If this public key remains associated with the particular device during the entire sales chain, it can be used to simplify the ownership change process (cf. Section 4.4).

[6] Mixing in data from a weak local RNG is possible, but will not significantly increase the attack effort.

[7] A particularly malicious manufacturer might even include a fuse bit, which after activation would hide itself and the private key access possibility.

4.3 Ownership Change: Release

A device first must be released by its previous owner, before it starts looking for a new home. This is best done by sending an explicit command to the device by the previous owner (which, on the first sale, is the manufacturer), called an *active release*. Fallback methods (*passive release*) should include a timeout (not having seen the HomeCA for a predefined period, e.g. a week). To some extents, this can be considered a variation of the Resurrecting Duckling security policy model [SA99], which describes a transient ownership relation of a device between multiple owners.

4.4 Ownership Change: HomeCA Discovery & Key Verification

A released IoT device, whether actively or passively, will search for a new home network. After connecting to a network, a device will first use Multicast DNS Service Discovery (mDNS-SD, also known by its implementations Bonjour and Avahi) [CK13] to discover the HomeCA. The question to be answered is how the IoT device connects to the network in the first place. Two possibilities exist:
(1) Probe an open network: This is easiest to implement, but will make it prone to networks set up as traps by adversaries trying i.e. to collect IoT device information.
(2) Connect to a WPA2-Enterprise (WPA2 EAP-TTLS, Extensible Authentication Protocol Tunneled Transport Layer Security Authenticated Protocol Version) network. 802.1X is used to authenticate a client in the network with an individual Master Session Key (MSK) for each session. The authentication phase is extended such that contacting the HomeCA in a limited fashion prior to having full network access is already possible during this phase [Sm12]. Later, connections to a WPA Enterprise network will present the HomeCA certificate to safely connect in EAP-TLS mode.
As long as a device has not been accepted and granted full access by a HomeCA (cf. Section 4.5), it will keep probing.
When a device public key has been delivered as part of the order, it may be pre-configured in the HomeCA, allowing later automatic joining as soon as this device comes within network range. This is of course the most comfortable and thus recommended operation.
Without public key delivery, the device will try to join the network and contact the HomeCA. The device will sit in this *unverified* state until the user has manually confirmed the device addition. In the simplest case, this is performed similar to the WPS Push-Button authentication [Wi14]. However, we envision a control application on the owner's smartphone, which displays device information (e.g., name, type, joining time) to verify it is actually the correct device. This application may then also be used for rights management.

4.5 Ownership Change: Key Update, Rights Management & Certificate Creation

The key currently associated with a device might be known to the manufacturer (cf. Section 4.1) and/or previous owner. Anyone knowing the key can directly impersonate this device or perform an undetectable Man-In-The-Middle (MITM) attack. Thus, reusing the same key is to be avoided. Therefore, on association, a new key is generated. To avoid accidental association with a foreign HomeCA, an initial integration of a device must be accepted within a User Interface (UI) presented to the user, i.e. on a smartphone app. Also, an IoT device will only accept a new HomeCA as its owner, if it has been previously released (cf. Section 4.3). Only if the user actively searches for devices, both known and unknown ones are shown. Otherwise unknown ones are hidden.
On a key update, any sensitive data on a device is also wiped, making them unavailable to the previous owner. This may include access rights to other devices, measurement data, or information received from other devices while with the previous owner [VA14].
Optionally, the owner, possibly through the HomeCA app, can assign rights this device should have toward other devices. This is represented as an Access Control List (ACL) [GPR13, Le84], possibly in matrix form, listing allowed operations per target device or device group as shown in column two, *Allowed function*, in Table 1.
The HomeCA signs the public key together with device information, validity period, and the optional ACL. This certificate is passed to the device.

4.6 Refresh: Liveness Verification

Liveness verification is initiated by the IoT device and used to state their availability to HomeCA, to regularly update the certificate with the connection information and optional ACL. It also allows IoT devices to reactivate the certificate-based relationship with HomeCA after a long period without interaction, e.g., during holidays that exceeds the time window t.

4.7 Service Discovery & Connection Establishment

Configurationless service discovery for HomeCA is done using DNS Service Discovery (DNS-SD) over multicast DNS (mDNS) [CK13]; widely known as *Zeroconf* or *Bonjour*. Because this solution does not protect the users' privacy, we include a privacy extension as presented in [KW14].
An initiator device A presents its certificate implementing a TLS or DTLS profile [TF16] to a target device B (with optional ACL signed by HomeCA), claiming authorization to access the device and perform particular operations.
As a result, each connection is secured by (at least three) layers of additional protection: (1) Device A can only be access other HomeCA controlled devices B when presenting a valid certificate. (2) With ACLs in place, the actual operations that A can perform on

Tab. 2: IoT device ownership change: Key knowledge

Entity	Manufacturer key known	Old device key known	Key delta	New device key known
Manufacturer	YES	NO	NO	NO
Previous owner	NO	PERHAPS	NO	NO
IoT device	YES	YES	YES	YES
New owner	NO	NO	YES	YES

B are positively enumerated. (3) Device *B* will only accept a subset of protocols which are considered secure by HomeCA. (4) The gateway (that could be HomeCA itself or a dedicated device, cf. Table 1) can translate between incompatible security protocols and provide additional content filters for insecure devices.
If gatewaying/proxying is not supported by the particular application layer protocol for the device, it can be emulated by TLS Server Name Indication (SNI) as used in Domain Fronting [Fi15].

5 Secure Key Update

As we have seen, ownership change for IoT devices (Section 3.2) is common, entropy may be hard to come by (Section 4.1), and — as they can work with sensitive data but are hard to control (Section 3.1) — should receive minimal trust.
While open adversities by the manufacturer may be rare, negligence or process errors are not uncommon: In the early days of networking equipment, it started with batches of network adapters with matching hardware addresses. Today, it is identical or low-entropy encryption keys or the stealing of these keys in the facility [SB15]. So, these initial keys should be minimally trusted, but any entropy in them should be kept.
Even though the new private key should only be known to the device, the HomeCA can be sure it includes any additional entropy from the HomeCA as part of the key update process. This is achieved through knowledge splitting (cf. Table 2), i.e. the parties (manufacturers, devices, and owners) only know the keys they are working with, such that no party knows the complete set of cryptographic keys of the communication system.
We will base the description of our secure key update algorithm on the following three entities: (1) Device d with a possibly non-unique ECC (Elliptic Curve Cryptography) key, that might also be known to a third party, i.e. manufacturer M. (2) Manufacturer M that could know the ECC key of d. (3) HomeCA, e.g. running within an owners Smart Home system and is a trustworthy entity that shall not know the key.
Device d is created by manufacturer M. d is flashed with its software during the manufacturing process. At this time, M provides the following keys that are stored on d: A public key $\{M_{pub}\}$ for the communication with M, and a unique public key infrastructure (PKI) key pair $\{d_{pub}, d_{priv}\}$. M stores $\{d_{pub}\}$ only but must be expected to know the private key, too.

HomeCA can verify for d, that update U is actually from M since the update is signed with the certificate of M. Access to d is only granted via HomeCA because device d only accepts messages encrypted with $\{d_{pub}\}$ which is known to HomeCA only. The protocol steps for the integration of d into the home network with HomeCA protocol are the following:
(1) A key pair k for device d including entropy from the previous key and additional entropy from the information exchange between the HomeCA and the device is created. (2) HomeCA continuously broadcasts its presence within the private network and requests new devices for authentication. (3) d authenticates against HomeCA with its key pub_d . (4) Auxiliary condition: Geographically restricted and within a small time window to reduce the attack vector. (5) d and HomeCA create a common Diffie-Hellman Key. The result is not predictable by both. (6) Entropy check: Verification of the quality of the random number r. (7) d creates a new key pub_{d2} and $priv_{d2}$, that are obtained by multiplication of pub_d and $priv_d$ with k. (8) HomeCA signs the new key pub_{d2}, that HomeCA can obtain from $k * pub_d$ and provides the certificate to d.
As a result the private key $priv_d$ of d is known to d and potentially also to the the manufacturer M. The key k is known to d and HomeCA, whereas $priv_{d2} = k * priv_d$ is only known to d. At the same time, HomeCA knows, that d has integrated material.

6 Conclusions and Future Work

This paper addresses the challenges of the integration of IoT devices into a private network. We present our mostly automated security protocol HomeCA. It focuses mainly on two functions: First, secure integration, which relies on PKI cryptography that prevents attacks from the Internet. Second, scalability to a large number of IoT devices, by the automated integration process without user interaction based on defined protocol steps within the private network.
The steps described in Section 4 ensure that on first purchase and later ownership changes, the keys are updated securely, even when the device lacks a reliable entropy source. The processes are designed to ensure long-term compatibility and security, even when it is expected that the devices will not be provided with security updates.
For the future it is envisioned to extend the implementation of the HomeCA protocol in an environment with more devices to study and analyze its viability without requiring significant changes on existing IoT devices. This full implementation will also allow deployment to verify several parameters, including the validity periods and timeouts.

Bibliography

[BJD16] Bugeja, Joseph; Jacobsson, Andreas; Davidsson, Paul: On Privacy and Security Challenges in Smart Connected Homes. In: European Intelligence and Security Informatics Conference (EISIC). IEEE, 2016.

[Bo14] Bongard, Dominique: , Offline bruteforce attack on WiFi Protected Setup. `http://archive.hack.lu/2014/Hacklu2014_offline_bruteforce_attack_on_wps.pdf`, 2014. (Accessed: 2019/06/17).

[BW15] Barcena, Mario Ballano; Wueest, Candid: , Symantec Secrity Response: Insecurity in the Internet of Things. https://tinyurl.com/yb227t7d, 2015. (Accessed: 2019/06/17).

[Ca17] Carnegie Mellon University: , CAPTCHA: Telling Humans and Computers Apart Automatically. http://www.captcha.net/, 2017. (Accessed: 2019/06/17).

[CK13] Cheshire, S.; Krochmal, M.: , IETF RFC 6763 DNS-Based Service Discovery. https://tools.ietf.org/html/rfc6763, 2013. (Accessed: 2019/06/17).

[Fi15] Fifield, David; Lan, Chang; Hynes, Rod; Wegmann, Percy; Paxson, Vern: Blocking-resistant communication through domain fronting. In: Proceedings on Privacy Enhancing Technologies. pp. 46–64, June 2015.

[GM94] Gander, Martin J.; Maurer, Ueli M.: On the Secret-Key Rate of Binary Random Variables. In: International Symposium on Information Theory. IEEE, 1994.

[GPR13] Gusmeroli, Sergio; Piccione, Salvatore; Rotondi, Domenico: A capability-based security approach to manage access control in the Internet of Things. Mathematical and Computer Modelling (Elsevier), 58:1189–1205, 2013.

[Ha16] Harmon, Elliot: , Don't Hide DRM in a Security Update. https://www.eff.org/deeplinks/2016/09/dont-hide-drm-security-update, 2016. (Accessed: 2019/06/17).

[IE13] IEEE Computer Society: 1149.1-2013 - IEEE Standard for Test Access Port and Boundary-Scan Architecture. In: Working Group: Boundary Scan Architecture - Standard Test Access and Boundary Scan Architecture WG P1149.1. 2013.

[Kl15] Klugah-Brown, Benjamin; Aristotle, John Bosco; Ansuura, Kanpogninge; Qi, Xia: A Signcryption Scheme from Certificateless to Identity-based Environment for WSNs into IoT. International Journal of Computer Applications (0975 – 8887), 120(9), 2015.

[KW14] Kaiser, Daniel; Waldvogel, Marcel: Efficient Privacy Preserving Multicast DNS Service Discovery. In: Workshop on Privacy-Preserving Cyberspace Safety and Security (IEEE CSS). IEEE, pp. 1229–1236, 2014.

[Le84] Levy, Henry M.: Capability-Based Computer Systems. Butterworth-Heinemann, 1984.

[LXC12] Liu, Jing; Xiao, Yang; Chen, C. L. Philip: Authentication and Access Control in the Internet of Things. In: 32nd International Conference on Distributed Computing Systems Workshops. 2012.

[Ma16] Maire O'Neill: Insecurity by Design: Today's IoT Device Security Problem. Engineering (Elsevier), 2:48–49, 2016.

[Mi18] Microchip Technology Inc.: , AWS Zero Touch Secure Provisioning Platform. http://www.atmel.com/applications/iot/aws-zero-touch-secure-provisioning-platform/default.aspx, 2018. (Accessed: 2019/06/17).

[MM15] Markowsky, Linda; Markowsky, George: Scanning for Vulnerable Devices in the Internet of Things. In: 8th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications. 2015.

[RM12] Rescorla, E.; Modadugu, N.: , IETF RFC 6347 Datagram Transport Layer Security Version 1.2. https://tools.ietf.org/html/rfc6347, 2012. (Accessed: 2019/06/17).

[SA99] Stajano, Frank; Anderson, Ross: The Resurrecting Duckling: Security Issues for Ad-hoc Wireless Networks. In: International Workshop on Security Protocols. April 1999.

[SB15] Scahill, Jeremy; Begley, Josh: The Great SIM Heist. The Intercept, February 2015. https://theintercept.com/2015/02/19/great-sim-heist/ (Accessed: 2019/06/17).

[Sc15] Sciancalepore, Savio; Capossele, Angelo; Piro, Giuseppe; Boggia, Gennaro; Bianchi, Giuseppe: Key Management Protocol with Implicit Certificates for IoT systems. In: 1st International Workshop on IoT Challenges in Mobile and Industrial Systems. 2015.

[Sh18] Shah, Rashmikant B; Weis, Brian E; Kumar, Kannan; Nayak, Manoj Kumar: , Zero-touch iot device provisioning, November 1 2018. US Patent App. 15/582,113.

[Si15] Sivaraman, Vijay; Gharakheili, Hassan Habibi; Vishwanath, Arun; Boreli, Roksana; Mehani, Olivier: Network-Level Security and Privacy Control for Smart-Home IoT Devices. In: 11th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob). IEEE, pp. 163–167, Oct 2015.

[Sm12] Smith, Tim: , 802.11 Sniffer Capture Analysis - WPA/WPA2 with PSK or EAP. https://supportforums.cisco.com/t5/wireless-mobility-documents/802-11-sniffer-capture-analysis-wpa-wpa2-with-psk-or-eap/ta-p/3116990, 2012. (Accessed: 2019/06/17).

[Su12] Suo, Hui; Wan, Jiafu; Zou, Caifeng; Liu, Jianqi: Security in the Internet of Things: A Review. In: 2012 International Conference on Computer Science and Electronics Engineering (ICCSEE). volume 3. IEEE, pp. 648–651, March 2012.

[Sv16] Svenda, Petr; Nemec, Matus; Seka, Peter; Kvasnovsk, Rudolf; Formane, David; Komarek, David; Matyas, Vashek: The Million-Key Question — Investigating the Origins of RSA Public Keys. In: 25th USENIX Security Symposium. USENIX, 2016.

[TF16] Tschofenig, H.; Fossati, T.: , IETF RFC 7925 Transport Layer Security (TLS) / Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things. https://tools.ietf.org/html/rfc7925, 2016. (Accessed: 2019/06/17).

[UC17] US-CERT: , Alert (TA16-288A): Heightened DDoS Threat Posed by Mirai and Other Botnets. https://www.us-cert.gov/ncas/alerts/TA16-288A, 2017. (Accessed: 2019/06/17).

[VA14] Vidalis, Stilianos; Angelopoulou, Olga: Assessing Identity Theft in the Internet of Things. journal of IT Governance Practice, Vol. 2 (1): 15- 21, 2014.

[VP16] Vanhoef, Mathy; Piessens, Frank: Predicting, Decrypting, and Abusing WPA2/802.11 Group Keys. In: 25th USENIX Security Symposium. USENIX, 2016.

[VP17] Vanhoef, Mathy; Piessens, Frank: Key Reinstallation Attacks: Forcing Nonce Reuse in WPA2. In: 24th ACM Conference on Computer and Communications Security. 2017.

[Wa16] Waldvogel, Marcel: , DDoS: What we can do to prevent it. https://netfuture.ch/2016/09/ddos-what-we-can-do-to-prevent-it/, 2016. (Accessed: 2019/06/17).

[Wi14] Wi-Fi Alliance: , Wi-Fi Certified Wi-Fi Protected Setup. https://www.wi-fi.org/discover-wi-fi/wi-fi-protected-setup, 2014. (Accessed: 2019/06/17).

[Zi15] Zillner, Tobias: , ZigBee exploited: The good, the bad and the ugly. https://www.blackhat.com/docs/us-15/materials/us-15-Zillner-ZigBee-Exploited-The-Good-The-Bad-And-The-Ugly-wp.pdf, 2015. (Accessed: 2019/06/17).