Journal of Reliable Intelligent Environments (2019) 5:41–64
https://doi.org/10.1007/s40860-019-00075-0

ORIGINAL ARTICLE

What it takes to boost Internet of Things privacy beyond encryption with unobservable communication: a survey and lessons learned from the first implementation of DC-net

Ralf C. Staudemeyer[1] · Henrich C. Pöhls[2] · Marcin Wójcik[3]

Received: 2 September 2018 / Accepted: 12 February 2019 / Published online: 26 February 2019

Abstract
Privacy requires more than just encryption of data before and during transmission. Privacy would actually demand hiding the sheer fact that communication takes place. This requires to protect meta-data from observation. We motivate the need for strong privacy protection in a smart home use case by highlighting the privacy issues that cannot be solved by confidentiality mechanisms like encryption alone. Our solution is a implementation of DC-net on Re-Mote sensor nodes running Contiki OS. From this, we conclude that the computational and network overheads imposed by these techniques do not make them impractical to use in the IoT. To the best knowledge of the authors, this is the first implementation of DC-net on sensors. Alongside, we provide a survey of the required strong cryptographic security mechanisms, like encryption of communication, to be in place. We describe how the current existing techniques can be facilitated to achieve unobservable communication for the IoT. This includes mechanisms for encrypted IoT communication like DTLS or message authentication like ECDSA signatures on IoT devices. For readers unfamiliar with the concepts of MIXing and DC-net, we explain and analyse how those techniques, formerly used to provide private communication in the Internet, can be applied to the IoT. We briefly survey what complementary features from the IoT architecture are helpful in providing strong protection in this particular use case. Finally, we state some recommendations hoping that following these will enable us to reduce the privacy invasiveness of the IoT on all levels. We think that this will be indispensable if IoT devices shall become a part of our daily lives without rendering us into an Orwellian society.

Keywords Internet of Things · Privacy · Security · Unobservable communication · Dining cryptographers

1 Introduction

Privacy in the Internet of Things (IoT) needs more than encrypted end-to-end communication, as meta-data still leaks. On the other hand, privacy is understood as a human right, the threat arising from meta-data collection and analysis must be countered with strong security features. The strongest security feature suitably and cryptographically realisable for local environments is unobservable communication.

Taking a step back, the reason that encryption, even end-to-end on its own does not give the user privacy is that meta-data is still being collectable. Even in overencrypted channels, an observer on the network might be able to determine the sender and receiver of packets[1] or the sheer size of packets [81][2], their frequency, or their timing in correlation

✉ Ralf C. Staudemeyer
r.staudemeyer@hs-sm.de

Henrich C. Pöhls
hp@poehls.com

Marcin Wójcik
marcin.wojcik@cl.cam.ac.uk

[1] Faculty of Computer Science, Schmalkalden University of Applied Sciences, Schmalkalden, Germany

[2] Institute of IT-Security and Security Law (ISL), Chair of IT-Security, University of Passau, Passau, Germany

[3] Computer Laboratory, University of Cambridge, Cambridge, UK

[1] Unless the communication is on a broadcast medium and the broadcast nature is also facilitated in the network protocol.

[2] For example, the authors of [81] state that "[...] we cannot identify DNS requests by their disclosed payload. We overcome this by identifying packets through their length: DNS packets are usually smaller than other TCP packets." [81]

to other packets and events leaks information. Of course far less than unencrypted channels, but as the Internet started moving to more and more encrypted communication, we see that the attackers consult the collected meta-data. So our assumption is that clearly M. Weiser's vision of ubiquitous computing [95] is becoming our reality and the so-called "smart" things are monitoring us directly or indirectly in our physical surroundings. And as devices, they can be installed even in our home [35], or in street lamps,[3] roads, public transport, and buildings.[4] We also argue that even though there are still way too many unencrypted channels in the IoT,[5] the challenge of a properly encrypted local IoT network's traffic or the traffic between an IoT device and its cloud back-end is technically already being solved, e.g., DTLS and ECDSA help IoT devices to communicate securely—but as we will highlight in this article: on its own encryption does not enable privacy.

When privacy is understood as a human right, then we must technically counter the privacy threat arising from meta-data collection and their analysis with strong technical security mechanisms. Data protection laws in the EU, i.e., Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data [89], require—among other things that personal data may be gathered only for a precisely specified purpose. More recently, the EU protection laws were boosted by the release of the EU General Data Protection Regulation (GDPR) [32] which came into force in all European member states in May 2018 and has higher fines. In April 2017, it was estimated that fines only within the UK would be 79 times higher under GDPR.[6] The EU-GDPR encompasses the principle of privacy-by-design [14,25] and they require to "minimize the amount of collected data" [31]. Further, it requires the individual person whose personal data is handled to give their informed consent a priori to the data gathering and must be able to intervene. As many embedded devices are sold as products inside the EU or to the governments, e.g., smart cities [85], the data protection laws play an essential role in particular for the IoT. We are yet to see how the GDPR will be used in case of privacy breaches due to IoT involvement. To not let it come to this point, we analyse in this work if the strongest security mechanisms suitably and cryptographically, which is unobservable communication, are realisable for local IoT environments.

1.1 The privacy problem of IoT

A human that is within the monitored space becomes the legal data subject and has legal rights to access, modify and demand deletion of personal information collected. The EU's report on privacy in the IoT released in 2014 [31] shows the increased sensitivity of the topic. The need and awareness within the EU that strong protection is required is also highlighted by the European Commission's support for projects like PANORAMIX[7] that aims to build a mixed networking framework similar to onion routing of TOR [26,77]. Those kind of networks already work towards very strong protection of the communication relation in low-latency environments using chained proxies with special properties. Thus, once you want to offer an IoT service or product in the EU, the data protection rights of the data subject need to be respected as they are guaranteed by EU law. However, also outside the regulatory domain of the EU, we strongly believe that it is a human right to have strong protection of personal data.

Having privacy means that you gain the ability to prohibit the leakage of information to unauthorised third parties. As a first step, encrypted and authenticated channels technically ensure that only authorised parties are able to read a message's payload during transmission. They found their way into standards in the IoT domain, e.g., Datagram Transport Layer Security (DTLS) [54,57] (see Sect. 5.2 for more details). However even assuming DTLS or the like is enabled, and Hewlett Packard's recent report on IoT security finds just the opposite, i.e., that "70 percent of devices used unencrypted network service" [40], meta-data still leaks details about the communication. It is very hard to estimate to what extent meta-data can be gathered and utilised by network traffic analysis. Among other things, meta-data includes information like communication endpoints, message timing and location details of the communication. When combined with a priori knowledge, and processed by machine learning algorithms, the extracted information can be so rich that end-to-end encryption can be bypassed. For example, it might not be necessary to decrypt the payload at all, because its content can solely be derived from network traffic. This strongly demands an additional layer of privacy protection to prevent the leakage of sensitive information from meta-data, namely unobservable communication (UC).

[3] https://www.tvilight.com (last acc. 09 January 2019).

[4] https://www.greenerbuildings.eu (last acc. 09 January 2019).

[5] A 2015 research by HP "Internet of Things research study 2015" [40] indicated that "90 percent of devices collected at least one piece of personal information". Furthermore the HP Internet of Things Security Study: Home Security Systems Report 2015 [41] reported that "100 percent of home security systems do not require strong passwords" and that "50 percent exhibited improperly configured or poorly implemented SSL/TLS".

[6] https://theregister.co.uk/2017/04/28/ico_fines_post_gdpr_analysis/ (last acc. 09 January 2019).

[7] https://panoramix-project.eu (last acc. 09 January 2019).

1.2 Introduction to unobservable communication (UC)

The key concepts of information security are confidentiality, integrity, and availability. Confidentiality and integrity can be addressed with signatures and end-to-end encryption using public-key cryptography. But irrespective from this, attackers can monitor communication and analyse network traffic. Traffic analysis focuses solely on communication patterns and the extraction of information out of meta-data.

To counter traffic analysis, we need to minimize any kind of information leakage due to communication meta-data and content data to the same extent. Therefore, the network communication property we aim for is unobservability. This property ensures that messages and random noise are indistinguishable from each other. In terms of network nodes, it ensures that their activity goes unnoticeable and that messages cannot be correlated. It is a very powerful property ensuring unlinkability, unidentifiability, and a continuous flow of dummy traffic.

$$\text{Unobservability} = \text{Anonymity} + \text{Dummy Traffic with}$$
$$\text{Anonymity} = \text{Unidentifiability} + \text{Unlinkability}.$$

Unlinkability ensures that neither messages nor network nodes can be correlated. Unidentifiability[8] ensures that these are indistinguishable, building a so-called anonymity set. Given the anonymity set is always > 1, the system provides the anonymity property. The appropriate use of dummy traffic then yields to a corresponding unobservability set. These terms are defined in great detail by Pfitzmann and Hansen and are discussed in [66].

1.3 Goals, contributions and organisation

"Truly smart gadgets should have built-in intelligence" as proclaimed by Tony Fadell, the inventor of Nest thermostats [92]. In this work, we show how to put this smartness inside the local devices of the IoT to a good use to increase privacy of the network. Note that other options exist to decrease the privacy invasiveness of many currently deployed or foreseen IoT deployments. However as we briefly discuss in Sect. 7, these require to be tailored to the data gathered or the processing results. The property provided by unobservable communication is beneficial to any application and any data, and resists any attacker observing local traffic. That such a rough network device is not some artificial attack scenario can be easily motivated if you assume a burglar trying to learn if a smart home is occupied. To not fall behind in the battle over privacy, this article uses the IoT devices' smartness to technically strengthen the privacy as good as possible for a very broad application domain. Hence, in this work we concentrate on the network layer and present how to build a network between IoT devices at the edge with very strong security and privacy properties.

Our contribution is twofold, first we survey and explain ideas and existing mechanisms for unobservable communication and present how they can improve privacy in the IoT, second we present an implementation of DC-net in Sect. 6 and give an estimation of the overhead that a truly private IoT environment would induce.

The remainder of this work is structured as follows:

- In Section 2, we present the current general state of affairs in IoT Privacy, state of the art is discussed when needed to aid readability.
- In Section 3, we define a specific smart home scenario to reduce the otherwise very broad and general term "IoT". We use it to motivate privacy problems and gains.
- In Sect. 4, we analyse existing, sound solutions for private communications based on existing cryptography that work in the Internet, and discuss their suitability for unobservable communication in the IoT. Also from the legal perspective, the IoT is not an exception from the challenges we face on the 'general' Internet, but the IoT is an exceptional case due to its ubiquity and thus its potential to intrude into peoples' private lives.
- In Sect. 5, we check the current available building blocks.
- Our new contribution is in Sect. 6: we implemented the DC-net protocol [17] on constrained node devices. We also present our first estimations of the overhead needed for our actual implementation of an unobservable network communication in the IoT.
- In Sect. 7, we look beyond our new defence against privacy on the network layer of the device-to-device communication offered by unobservable communication and we discuss the freedoms that are required in the architectural frameworks of IoT and to enable privacy from the device to the application.
- We conclude in Sect. 8 and give some recommendations in Sect. 9.

2 Current state of the IoT towards Privacy

Bandyopadhyay and Sen [4] identify security and privacy as key technologies that will enable IoT. Privacy of humans and confidentiality of business processes are identified as the two major issues in this regard. Confidentiality can be adequately addressed with standard encryption technology, provided there is a suitable key distribution scheme available. More effort is needed to increase performance and at the same time lower power consumption of IoT devices. The

[8] A stronger property than unidentifiabilty is undetectability, where the attacker cannot distinguish if the item of interest exists or not [66].

authors point out that there is a lack of privacy preserving technologies available for IoT environments. They identify anonymity networks as a potential basis to implement privacy in IoT. This technology adds significant complexity and resource requirements in terms of computing power and bandwidth.

The work published by Miorandi et al. [56] provides a survey on the key issues related to the development of IoT services and applications. The authors identify data confidentiality, privacy and trust as key challenges in IoT security. Healthcare applications, the excessive use of wireless network technology and remote access capabilities are given as examples. Those technologies potentially expose very personal data to eavesdropping. The authors suggest addressing privacy issues in the system design phase, but admit the lack of a privacy framework tailored for the IoT.

An analysis of security challenges in distributed IoT environments is provided by Roman et al. [79]. The authors argue that potential threats and attackers need to be modelled first. Considered are denial of service, physical damage, eavesdropping, node capture, and controlling of IoT entities. Security challenges identified and discussed are identity and authentication, access control, protocol and network security, privacy, trust management, governance, and fault tolerance. The authors conclude that the heterogeneous nature of IoT increases the complexity of most security mechanisms.

Nevertheless some mechanisms might also benefit, like fault tolerance and trust, thus facilitating the implementation of privacy-by-design principles documented by Cavoukian in [14]. When misused, the IoT might as well turn into a security nightmare and worsen privacy issues by allowing even better user tracking and profiling.

This need for privacy(-by-design) is acknowledged by the European Union (EU) [25]. The EU Article 29 Working Party released a list of recommendations to increase privacy of IoT deployments [31]. Among other things, they recommend that "Device manufacturers should limit as much as possible the amount of data leaving devices" [31]. The latter is referring to the minimisation of information inside the payload, and they suggest aggregation of data.

The EU-funded project RERUM[9] that developed a framework will allow IoT applications to consider security and privacy mechanisms early in their design phase. This works towards a configurable balance between reliability (requiring secure, trustworthy and precise data) and privacy (requiring data minimisation for private information, like location) [70,85,91]. All EU-funded projects working on security and privacy in the IoT (IERC AC5) reported jointly in January 2015 that there is currently no research project that tackles anonymising the traffic in networks used for IoT applications [3, p.70]. This was picked up by [5,86] and is extended in this article. We show that it requires network security, changes in the way devices communicate and above all it requires that the IoT architecture is flexible enough to accommodate the new requirements without having to overhaul the whole architecture.

3 Traffic observations and their privacy problems examplified

In the following, we are going to describe a smart home scenario. The reasons for the choice of this scenario are manifold. First, it is an IoT scenario that is in its effects directly visible in our daily lives. Second, it is most likely to happen on a broad scale in the near future and thus affects many people—even if they are early adopters and have only a small number of connected devices in their homes. Many products, e.g., "smart" light bulbs or connected loud speakers, are already sold in some quantities as of 2018 today.

Development frameworks are maturing, for instance the operating system Contiki[10] or Android Things[11] or visions have been expressed.[12][13] Third, it has also been selected as a scenario by research projects in the smart city domain, e.g., RERUM or CITY PULSE.[14] Last but not least, it allows us to highlight privacy problems in a comprehensible manner, e.g., the direct real life implication of a breach of privacy is clearly visible when assuming a burglar as the attacker.

3.1 Smart home use case description

This section will stay informal to highlight certain privacy issues that can only be addressed by anonymous or unobservable communication systems. In the remainder of the paper, we will state more formally the specific attack model(s).

It is obvious that IoT devices will gather information about the home's inhabitants. Data protection regulation says that if the information is relevant to the home owner's privacy it must not be shared with third parties, unless the home owner has given informed consent, assuming the owner being the sole data subject. While this is indeed still a problem to be solved, it can be partially addressed by encryption and entity authentication, e.g., using a public-key encryption scheme. To protect confidentiality, for instance, a sensor could encrypt

[9] https://ict-rerum.eu (last acc. 09 January 2019).

[10] https://contiki-os.org (last acc. 09 January 2019).

[11] https://developer.android.com/things/index.html (last acc. 09 January 2019).

[12] Corning's Day Made of Glass https://youtube.com/watch?v=6Cf7IL_eZ38 (last acc. 09 January 2019).

[13] Panasonic's Wonder Life-BOX 2020 https://panasonic.com/global/corporate/center/tokyo/floor/lifebox2020.html (last acc. 09 January 2019).

[14] https://ict-citypulse.eu/scenarios (last acc. 09 January 2019).

readings with the designated server's public key, such that the data can only be decrypted by this selected server, assuming the data owner gave his/her consent to share this information to that server. Our motion detector could encrypt the "detected_motion" message for the smart home gateway, which in turn would encrypt and sign the command "turn_on" for the light. However, encryption of the payload does not prevent activity within the house from being observable. For instance, the detection of movement of the homeowner is observable for an attacker eavesdropping on the communication.

To monitor messages within a house or vehicle, the attacker needs to eavesdrop on a user-controlled network of sensors, actuators, and the gateway. This assumes that messages are not traversing non-private access networks, like DSL, 3G or the public Internet in general. The attacker needs to access the local network. In Fig. 1, the attacker would either need to be near the house to eavesdrop the wireless transmission, or the attack could be carried out by a trojan device (e.g., an attacker that controls a malicious sensor).

However, if "smartness" requires data to be sent to the Internet, for example, to request the weather forecast, then the attacker could eavesdrop at many locations. In our smart home example, this would be requesting the weather forecast. Point of observation could be not only the sensors, the local network, or the access network, but also the Internet, or the servers providing the requested service.

To stay within our specific example: observation of the in-house message from the motion detector, as well as the observation of the request sent to the weather forecast service or the TTS service over the Internet, will leak at what time you get up. Observation of communication flows, as an attack on privacy, is called traffic analysis. The event-driven nature of the communication flows in the IoT also lends those messages to be a good basis for attacks that facilitate pattern analysis. For example, the knowledge that the lights might receive their messages (containing commands) from different sources (switches and movement detectors) make them become recognised as actuators. The messages that relate to "turn lights on", "get weather forecast" and "translate to speech" always follow in the same order, and occur most likely within a specific time window each day. Therefore, at least the weather and text-to-speech service provider will know when you get up every morning. All this makes the IoT messages, even if their content is encrypted, a very rich hunting ground for meta-data analysis.

3.2 (Resisting) network traffic analysis

Traffic analysis [23,76,87] is the process of capturing network traffic and analysing it. The aim is to gather information about the network and its devices from observed communication patterns. This technique works without knowledge of the contents of the communication. Hence, it is also applicable to encrypted messages. Large numbers of captured messages make traffic analysis more effective. It might reveal even more information if it were possible to modify either the traffic flow or the messages themselves.

Information extracted with traffic analysis on its own does not seem very useful, but in combination with a priori knowledge, statistics and machine learning algorithms it becomes a very powerful tool. The mere fact that a couple of devices exchange messages might not be all that interesting on its own. The situation changes dramatically when we can map devices to locations, device types, and communication patterns to services.

For instance, if we guess that one communication node is a motion detector and the other is a light, then we can deduce further information. Observing network traffic, we are able to tell where these devices are located in the meshed network. Additionally, we can conclude that the sensor pushes a message most probable only on motion activity.

An eavesdropper can log frequency and time of all messages between two communication partners. The analyst can easily conclude whether the session is interactive or non-interactive. The motion detector firing whenever there is motion, is an example of a non-interactive session. If the light asks another sensor for the current ambient light level, so it does not turn on at bright daylight, then we call this an interactive session. The traffic frequency pattern of a non-interactive session is much more regular than that of an interactive session. Much like a fingerprint, an interactive session will have a very individual traffic pattern and therefore further deductions can be made.

Again, thinking of just the traffic pattern, motion sensor activity caused by the movements of a human or a pet differs; activity patterns between the usage of the same apartment by different people will also likely differ. Therefore, we must assume that the sensor's network message pattern reveals private data.

3.3 Observing the use case in more detail

Let us observe the "meshed" network of nodes of our smart home use case in Fig. 1 more closely. Fig. 1 pictures a subset of its sensors and actuators. There is an environment sensor, two activity sensors, and one light bulb as an actuator. The environment sensor sends its measurements automatically to the smart hub in a 30 s interval. The two other sensors detect movement and noise in the house. All devices use the IEEE 802.15.4 LR-WPAN standard for low-power wireless communication. All messages are forwarded to a smart hub processing and acting in accordance with their content.

Once the motion activity sensor gets activated, it sends a "motion detected" message to the smart hub. In the smart hub, this triggers an internal timer, here for example, for 2 min.

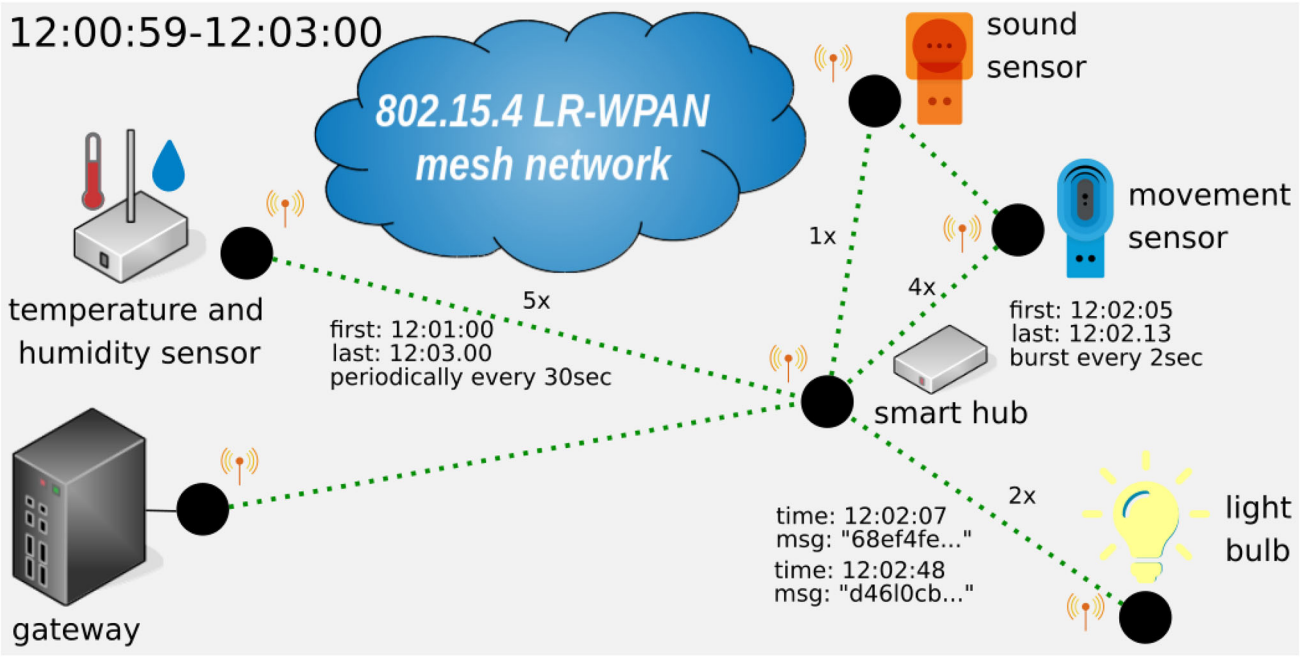

Fig. 1 Example of traffic between the devices in our smart home use case

As a consequence, the hub sends a "turn-on" message to the light bulb. This message causes the light bulb to switch into its "lights-on" state. On lasting activity, the motion sensor keeps on sending detection messages in burst intervals, for example, every 2 s. Every of these "motion detected" messages causes the smart hub to reset its timer. Assuming the motion sensor detects activity in intervals less then 2 min, the timer of the hub resets frequently and therefore remains active.

The sound sensor shows a similar activity: once it detects noise it sends a "noise detected" message to the smart hub causing the timer of the hub to reset correspondingly. The timer in the smart hub will remain active unless it does not receive any "detected" messages from one of the two activity sensors for the predefined time period. Here the timer will time out after 2 min which triggers to send a "lights-off" message to the lamp causing it to switch off.

Next we look into traffic analysis assuming all messages are end-to-end encrypted. Then we can make the following first observations:

1. three nodes have only outbound traffic
2. one node has only inbound traffic
3. one node is sending and receiving messages

Observations give us basic information about the roles of the nodes. We conclude that the three nodes with outbound traffic are most probably sensors. The one node with only inbound traffic is most probably an actuator. The sending and receiving notes might be a smart hub or some sort of gateway.

Now we can add simple message size, timing and frequency analysis to gain additional information.

- one sensor sends messages with a distinguishable, larger message size periodically with a 30 s interval
- two sensors send messages in a specific, reoccurring order with partially a 2 s interval
- two sensors and the one actuator have similar message sizes of two types each
- two sensors and the one actuators activity are correlated
- one sensor seems to trigger this ordered activity

For more information, we can create a database with fingerprints of communication activity of various sensors. We estimate that chances are high that we can clearly identify the environment sensor due to the larger message size. For the two activity sensors for motion and sound, we estimate good chances that we can identify them based on message size and timing. Identifying more details on the smart hub might require long-term observation.

First thoughts are: to prevent message size and frequencies from revealing information about the connection's type, it is necessary to exchange messages with a constant size and a rate flow with all potential communication partners. This is achievable by broadcasting a continuous stream of indistinguishable real and dummy messages, looking equal randomly to all parties except to the sender and the recipient(s).

However, this is not very efficient.

Next we will consider the different types of attackers.

3.4 Attacker models considered

In this section, we describe the building blocks for an attacker model based on ideas communicated in [87]. We briefly restate them for increased readability. The best would be to have *perfect protection* that even protects against an omnipotent attacker. This kind of attacker is able to trace all data from point of creation to delivery, alter all data unnoticeable, and alter whole system's functionality until demolition. However, this comprehensive protection against attacks by an omnipotent attacker is unrealistic to be fully gained. With respect to our use case, a realistic attacker model must:

- consider all possible attacks that can be expected during the lifetime of the observed system,
- be as simple as possible,
- be affordable to create and operate, and
- describe and point out the remaining risks.

Passive vs. active attacker

An attacker who is only able to eavesdrop on the communication medium and observe the traffic flow is defined as a passive attacker. It is not possible to recognise a passive attack. Nevertheless, it may be possible to recognise the success of a passive attack indirectly if such an attack changes the behaviour of nodes who collect additional information as consequence of an attack. We cover passive attacks in greater detail in Sect. 3.5.

An active attacker modifies network computations and transmitted messages. There may be restrictions on the kind of modifications an active attacker can execute. We can distinguish between interruption, interception and modification. Interruption cuts the connection between sender and recipient, e.g., an active adversary on the Internet could prevent the SmartHome gateway from reaching the weather service. With interception, an attacker is able to filter and store single packages. Modification enables an attacker to modify and introduce new packets, e.g., the weather service's answer might always be corrupted. We might further need to distinguish adversaries by their capability to modify packets in real time or only with noticeable delay. The attacker might also break the rules of the communication system. A modifying attacker may use protocols in ways which are not intended or even not expected in the network.

Attacker distribution and locality

An attacker may be able to control a subset of nodes of a communication system. These may be either network infrastructure devices like gateways or end devices like sensors. Control may vary from one to all available devices. Depending on the distribution of the attacker-controlled nodes within the network infrastructure and their relevance for the infrastructure, an attacker can obtain a more or less complete view of the overall network traffic flow.

We can define two special cases of attacker distribution: the global and the local attacker. A global attacker is omnipresent. He is able to access and observe all communication lines within the system. Even if an omnipresent global attacker is not considered realistic, other network participants might be corrupt and collude with the attacker. This way the attacker might be able to emulate an omnipresent attacker.

A locally present and/or an internal attacker has physical access to infrastructure devices. This may be a sensor, a gateway, an Internet service, or any number of intermediate nodes. Of special interest are intermediate nodes that provide routing or enhanced security functions. Nodes may be physically damaged or modified. It is important to consider that an attacker may be able to control a subset of available communication lines or intermediate nodes. In Fig.1, an attacker controlling the "Sensor Motion" and the "Actuator DoorOpener" will be able to control all the traffic of the 802.15.4 mesh network from and towards the gateway. An attacker controlling the 'Actuators Speakers' will most likely see less traffic passing by due to the position in the network. Hence, the strength of the attacker may vary strongly depending on the controlled subset of network participants.

Resources, adaptivity and non-technical attacks

The computing power of an attacker may be limited or unlimited. An attacker with unlimited computing power is known as an information theoretical attacker. It is risky to assume that an attacker has only limited hardware resources at his disposal or lacks knowledge of powerful algorithms. It is acceptable, however, to make assumptions on the attacker's maximal considered strength. This includes the amount of money the attacker is willing to spend on an attack. Time restrictions can also be assumed.

In a static attack, the compromised resources are fixed after the attack has been launched. An adaptive attacker is able to control and modify resources during an attack. Only adaptive attackers are able to trace messages. An attacker may also persuade a trusted third party to manipulate part of the network infrastructure. This could give additional power and capabilities to an attacker that would not be available under normal circumstances.

3.5 Eavesdropping—passive attacks

A passive attacker is only able to eavesdrop on communication links and intermediate nodes. It is the nature of attacks based on eavesdropping that they are not detectable as they are occurring. But it is possible to prevent their usefulness if the weaknesses of the observed network are known. In the

following, we give abstract descriptions of passive attacks that are independent of any specific concept and hence are most likely also applicable to the IoT domain [76,87].

Message timing

A timing attack observes the duration of communication between nodes and attempts to correlate patterns of network participation. Possible routes can be calculated using the round trip time of message sets entering and leaving the network at two observed points. Messages leaving a node A at time t and messages received/forwarded at time $t+RTT/2$ can be correlated with a predictable probability.

Message coding, size, counting, and volume

All messages that do not change their encoding or size during network transmission can be linked or traced [16]. A counting attack observes the number of packets exchanged between two possible communication partners. A combination of attacks based on tracking messages by size and number are communication volume attacks. They detect the communication relationship by observing the amount of transmitted data.

Communication pattern and message frequency

Communication pattern attacks can be launched by simply monitoring communication activity on any network device, i.e., the pattern and timing of message transmission and reception. Generally, an attacker can make the assumption that participating nodes usually do not send and receive messages at the same time. Observations over long periods of time can reveal sets of possible communication partners.

For message frequency attacks, an attacker analyses the traffic flow of messages to fingerprint individual devices and/or communication partners. This is most effective for real-time interactive communication. The attacker determines the message frequency between two endpoints by counting packets and recording the communication pattern. Communication pattern and message frequency attacks are challenging to overcome. They are most effective for real-time and interactive communication, both of which we find in IoT traffic.

Tracing individual packets

An attacker may trace every possible path an observed message can take through a network. From this, they can construct a list of possible recipients. Given enough time, the number of possible sender and recipient pairs can be reduced. In the worst case, an attacker can finally match one sender to one recipient. In the best case, an attacker needs to follow t^d messages along $t^d - 1$ paths through a network and to their recipients to identify a pair sender/recipient.

Long-term intersection attacks

An attacker may trace devices' network usage over a long period of time. Thus, it might identify them by their individual, characteristic usage pattern of network services. For instance, devices might exhibit specialised behaviour to contact the vendor, i.e., at vendor-specific regular intervals, for security fixes, or firmware updates.

In the next section, we discuss countermeasures.

4 Private communications

In this section we introduce concepts to counter passive attacks ([23,76,87]) based on eavesdropping and traffic analysis. We further set the context in the context of Internet of Things (IoT), private area networks (PANs), wireless sensor networks (WSNs), and our use case described in Sect. 3. In [85], we discuss these technical communication mechanisms towards privacy in a SmartCity context.

4.1 Basics to counter network traffic analysis

To ensure private communication, we need to consider countermeasures to the following passive attacks. Here we present an overview of relevant attacks and set the context to our use case.

To provide protection against message coding attacks, the encoding must change during transmission. This can be done by encrypting messages with k-nested layers to other members of the network. On the network side, this can be solved using per link encryption between routing nodes. A predefined message size of all network messages can protect against message size attacks; smaller messages must use padding at the cost of efficiency.

The propagation of messages needs to be randomly delayed. The minimum delay can be defined by the maximum possible latency for a communication. Otherwise, the attacker may be able to deduce that the message did not take a route with greater latency. The maximum delay should be as large as it is possible without interfering with real-time requirements.

Dummy traffic is most probably needed to provide sufficient protection. That means the motion detector would send messages even though no actual motion was triggered. These dummy messages must not differ in size from the actual event messages. Otherwise, the observation of traffic at network points close to the communication partners may reveal the communication relation. Prevention of attacks based on message counting is possible if all network participants send and

receive a standard number of messages. To provide protection against communication volume attacks, it is sufficient to protect against message size and message counting attacks.

Communication pattern attacks are dangerous attacks that are difficult to prevent. They require continuous network participation of a sufficiently large number of nodes. Message frequency attacks are most effective for real-time and interactive communication. A standardized, rigid message exchange pattern within the IoT network would help to provide enhanced protection.

Still, no guaranteed protection exists against brute force attacks or long-term intersection attacks. Keeping the number of possible recipients large at all times may make the success of such an attack less likely. This can also be achieved by introducing dummy traffic. Protection against long-term intersection attacks remains a well-known open problem. Continuous connectivity and message exchange with the network might be options to address this problem.

We conclude that protection against passive attacks requires a very rigid structure of communication. Generally, it remains challenging to overcome timing coincidences without wasting significant bandwidth. So far anonymity networks offer partial protection against such attacks since neither nodes or nor messages are easy identifiable.

4.2 Existing concepts

A very limited degree of anonymity can be achieved using a proxy or a virtual private network (VPN). An observer with access to traffic entering and leaving the proxy over extended periods of time can reveal the communication relation [16]. VPN networks on the other hand are slightly more robust, since they provide a layer of encryption implemented with secure communication protocols generally based on either IPSec or SSL/TLS. Therefore, incoming and outgoing traffic can not be mapped easily, but message frequencies and flow can still be analysed. Therefore, these solutions fail against a global observer. Fortunately, the situation improves using specific proxy chains. These tunnel encrypted traffic through a number of low-latency proxies.

There are a few basic concepts that offer adaptability to perfect protection against a global observer. One must distinguish between sender, receiver and mutual protection. In this context, mutual protection guarantees that both parties of a communication remain anonymous to each other and to any third party. For example, to broadcast or multicast, a message would allow recipient anonymity. A summary of anonymous communication systems is provided by [50]. The basic concepts that describe mechanisms for sending messages anonymously are MIXing [16], DC-net [17], and Anonymous Message Service [19].

We provide details on Mixing and DC-net, the basic concepts behind most systems, in the following: MIXing generates a high degree of anonymity and unlinkability of sender and receiver. It takes a branch of messages and scrambles, delays and re-encodes them in a way an attacker can no longer easily match incoming with outgoing messages. Several adaptations of the MIX concept were introduced, they add new functions and correct security problems. To obtain the unobservability property, an adaptation of the MIX concept was introduced in [67]. It includes constant Dummy Traffic and Time Slices [67] preventing an attacker to obtain useful information from packets travelling through the network.

David Chaum introduced in [17] the Dining Cryptographers Net (DC-net)—a communication protocol that provides unconditional secure unobservable communication. DC-net is a broadcast round-based protocol where members of the round can unobservably publish a one bit message per round. This is called "superposed sending" and is very secure but prone to denial of service attacks. By "superposed receiving" [94], DC-net was extended to support anonymous receiving of messages. Protections against disrupting nodes were proposed in [94] and [37]. Further, it is possible to categorise concepts into re-routing-based and non-re-routing based concepts [39]. DC-net and Broadcast (or Multicast) are the only non-re-routing based systems.

In practice, we need to consider the attack model and distinguish concepts to protect communication in two different environments: privacy area networks (PAN) and wide area networks (WAN). The attacker model for this scenario is a global attacker in a local environment, who is computationally bound, but can listen to every communication and can insert arbitrary messages, which is very realistic in a locally limited, wireless scenario. However, denial of service attacks, such as through radio jamming, are not considered. Given an attacker with powerful enough radio equipment, who is jamming radio frequencies used by the sensor nodes, there is very little chance of preventing this kind of attack.

In the following the concepts of Mixing and DC-net are roughly outlined and compared.

4.3 Mix systems—details

Anonymising proxy networks have started with the implementation of Chaum's Mix in 1981 [16]. The system tunnels encrypted traffic through a number of low-latency proxies. Initially, interest in this field was primarily theoretical but in the last 35 years a lot of research in this field has looked at developing practical and usable systems for preserving anonymity. There are a large number of existing Mix network implementations. Such systems cover Email ([24,60]), Web browsing and other services like peer-to-peer networks, IRC chat and hidden services [26].

Mixes offer a high level of sender-controlled protection for the sender and the receiver of messages. Mixing possesses several strong attributes. It can

- hide the relationship between sender and receiver,
- guarantee pseudonymity of sender and receiver,
- guarantee anonymity between sender and receiver
- guarantee anonymity of sender and receiver to all,
- protect against revelation of signalisation relations, location updates, time of communication, and kind of service.

Mixes receive asymmetric encrypted messages from different sources and put them in a queue. They delete replays, collect and decrypt messages and if a certain amount of messages are in the queue, they are pushed all out, but in a rearranged order. Initially the user encrypts his message m with an asymmetric method using the public key K_e of the receiver. Then the message is combined with a random part r_1 and encrypted with the public key K_1^{-1} of the MIX used to send the message.

$$m = K_1(r_1, K_e(r_0, m))$$

The random part in the message is necessary to prevent that an attacker can re-encrypt the outgoing message with the known public key of the mix and trace back the message. Additionally, outgoing messages all have the same length by adding random bytes and an additional layer of encryption.

$$m = K_1(K_1(r_1, K_e(r_0, m)), RandomBytes)$$

Multiple such MIXes must be used in combination, to increase security and to prevent that the owner of a single MIX can reveal the communication relation. The sequence is fixed by the encryption of the message using the individual public keys of the mixes. A message m using three mixes gets encoded in the following sequential manner.

$$K_1(r_1, K_2(r_2, K_3(r_3, K_e(r_0, m))))$$

Every MIX can only encrypt the outer frame of every message with its private key. Using multiple MIXes, it is sufficient if one of them is trustful. Just if all MIX carriers collaborate the communication relation can be revealed. Multiple MIXes can be organised in fixed cascades and in free routes. Within a cascade, a row of dedicated servers join and redirect traffic of a great amount of users down a predefined route. Fig. 2 shows an example of a Mix proxy node.

In the following section, we describe the concept of DC-net in greater detail, since it is more suitable for the PAN and WSN network environments we consider in this article.

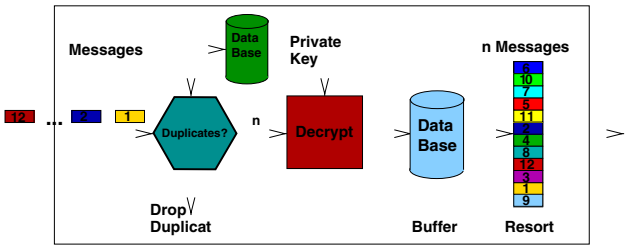

Fig. 2 Example of a Mix proxy node

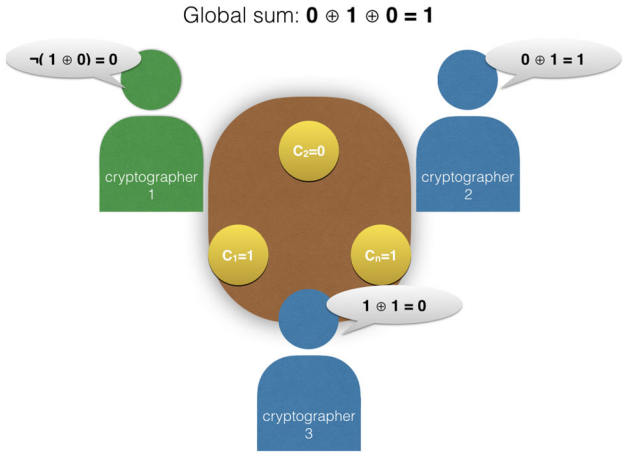

Fig. 3 One of the cryptographers has paid (green). Every cryptographer tosses a coin and shows it to his right-hand neighbour. Then, he computes the XOR of his coin with the coin on his left-hand neighbour. The cryptographers who did not pay announce this result, the cryptographer who paid (if any) announces the inverse. Here: $0 \oplus 0 \oplus 1 = 1$ which shows that one of the cryptographers paid and not the NSA

4.4 Superposed sending and DC-networks

The protection of the sender gets more challenging when the attacker controls most of the network. A message must get into the network to be delivered to the recipient at some point. The concept of "superposed sending" was developed to aid this process. With superposed sending and receiving, it is possible to build the DC-net [17,94], solution for the dining cryptographers problem [17].

Chaum introduced the DC-net with a little story: three cryptographers were having dinner in a restaurant and finally someone paid for the dinner—either one of the cryptographers or their employer, the NSA. Now, they want to find out who paid the bill, either one of them or the NSA. In case one of the cryptographers paid, however, they do not want to reveal who exactly. To solve this problem, they conceived the following protocol:

The cryptographers are sitting at a round table (depicted in Fig. 3) and each one of them is flipping an unbiased coin secretly and show the outcome to his right-hand neighbour. We refer to heads with "1" and to tails with "0". Now, each cryptographer combines the coin toss on his left and the one

on the right (the coin he tossed) with the XOR operation. A cryptographer who was not paying the bill, writes the result on his napkin in front of him, whereas a cryptographer who paid the bill, writes the negation of the result. The cryptographers see the local results from the others, and combine these results again with XOR, which we are referring as the global sum.

If, and only if, one of the cryptographers paid and is thus inverting the result of his local XOR operation, will the global sum be equal "1", since there are now an odd number of ones. Assuming that the coins were unbiased and no pair of cryptographers are colluding, this protocol is unconditionally secure.

In case the NSA paid, the global sum is "0" and there is no problem with anonymity. In case one of the cryptographers paid, it is equally likely for one cryptographer who wants to investigate the payer, that his left or right neighbour was paying. The XOR operation plays an important role here (later it will be explained, that also other mathematical structures can be used), since every coin toss $C_n, n \in 1, 2, 3$ exists twice in the global sum. Due to the commutativity and associativity of the XOR operation, the individual coin tosses,

$$\begin{aligned} &Local_{crypto1} \oplus Local_{crypto2} \oplus Local_{crypto3} \\ &= (C_1 \oplus C_2) \oplus (C_2 \oplus C_3) \oplus (C_3 \oplus C_1) \\ &= (C_1 \oplus C_1) \oplus (C_2 \oplus C_2) \oplus (C_3 \oplus C_3) \\ &= 0 \oplus 0 \oplus 0 = 0 \end{aligned}$$

cancel out in the global sum, which will always yield "0" as the global result. Only if one cryptographer who wants to send a (1-bit) message inverses his local result, will the global sum equal "1".

The DC-net protocol is unconditionally secure and because for the every cryptographer it is equally likely that either his left or right neighbour [17] paid the bill which permits participants to anonymously broadcast one bit messages. The protocol implicitly assumes the exchange of a one time pad over a secure channel and a reliable broadcast medium (cryptographers announcing and receiving the results without disturbance). In reality, those guarantees are hard to achieve but different optimisations like tackling the problem of reliable broadcasts, successfully addressed them and were presented in [94], [93]. Other optimisations make DC-net traffic unobservable by continuous communication of all participants using dummy traffic achieving authenticity, integrity and confidentiality with public-key cryptography instead of one time pads [76].

Key exchange, ring topology and dealing with disrupters

Coin tossing and sharing the result with the neighbours practically means exchanging secrets through a secure channel. For transmitting longer messages, many keys can be tossed beforehand and the outcomes can be shared. If the participants of a DC-net share secrets through a unconditional secure channel, the DC-net provides unconditional security. If key exchange is done through a public-key cryptography system, the security of a DC-net reduces to the degree of computationally security (relying on the security of public-key cryptography). The anonymity of a single node in a DC-net is expressed as the anonymity set for this node, that is the set of honest nodes which the node is sharing secrets.

Chaum proposes a ring as a possible network topology, that has compared to a traditional ring where messages often only travel have through the ring before the recipient gets them, a fourfold increase in bandwidth. To reduce bandwidth, not the whole message should be redirected by each ring node, instead, every node can build the XOR of his message and keys with the message he got from the previous node. This way, one global broadcast round has to travel twice through the ring to be received completely by the participants: in the first round every participant forwards the incoming message together with his local output and in the second round, the global sum is broadcasted to all members.

Another problem Chaum points out, is the disrupter's: malicious nodes, or nodes with faulty behaviour can disturb the whole communication by sending, for example, random data and thus making the global message unreadable. DC-net provides untraceability—also for those attacker—which makes it hard to detect them. Chaum proposes a trap mechanism, in which honest sender reverse a sending slot but instead of sending actual messages, they place traps in it by sending a random message with a secret key. If the attacker tried to disrupt the communication in this round by sending in the reserved slot of another node, the honest node that placed the trap detects this and signals this to the other participants together with his secret for this round and his decrypted message.

Reliable broadcast assumption

Waidner and Pfitzmann generalize and improve the concept of DC-nets [94] [93]. They generalize the concept of sender untraceability of superposed sending. The set of participants $P = P_1, ..., P_n$ are connected through a graph G. Instead of relying on the XOR operation, any Abelian finite group $(F, \oplus)$ can be chosen, which is called "alphabet". Participants share common secrets, that is, participant P_i and P_j share the secret key $K_{i,j} = K_{j,i}$. G denotes the key sharing graph, containing all nodes who shared secret keys with each other. Then each participant P_i chooses a message character M_i: 0 or the actual message if it wants to send (and owns the current slot). Then, the message M_i is combined with all keys shared with each other in this round. The $sign$ function

is used to create the vanishing effect (like $x \oplus x = 0$ with the XOR operation).

$$O_i = M_i \oplus \sum_{P_i, P_j \in G} sign(i - j) \cdot K_{i,j} \quad (1)$$

This local output of one node is then broadcast to all other nodes, which affords the other nodes the opportunity to compute the global result S of the current messaging round:

$$S := \sum_{j=1}^{n} O_j \quad (2)$$

Finally all keys occur twice in the sum with opposite signs, which leads to the vanishing of all keys and empty messages. Only the message with $M_i \neq 0$ is visible to all participants while preserving the anonymity of the sender.

Reservation map technique

Waidner also generalizes the reservation technique described by Chaum to additive groups of integers modulo m. Each participant who wants to send, chooses a position randomly in a reservation vector with r slots, and puts "1" into this position. After broadcasting those vectors, each participant can sum the positions up: "0" in a position means no one reserved this slot, "1" means one participant successfully reserved this slot without collisions and $n > 1$ means more than one participant wants to send within the timeframe of this slot. All slots with collisions are skipped and only successfully reserved slots are used by the specific participant.

Implementations

We are aware of only two DC-net implementations, Herbivore [36] and Dissent [20,96], probably due to DC-nets' sensitivity to disruption. However, none of them is aimed towards the IoT and constrained devices.

4.5 Overhead

Communication systems that provide strong security properties like anonymity and ideally unobservability suffer from a high computational and communications overhead. Already the initial key distribution problem, which requires keys to be exchanged with all potential recipients, makes it very expensive. In MIX networks, this concerns MIX nodes only, in DC-net this concerns all network participants.

The computational and bandwidth overhead in MIX networks is related to the threat model. For increased protection, the number of chained mixes and amount of dummy traffic need to be increased accordingly. A chain length of three running under different entities should make the correlation of sender and receiver sufficiently challenging. The computational overhead is therefore approximately $3 * k$, with k being the number a MIXes per chain. In terms of network traffic, the additional header for every nested message causes on overhead of $8 + 8 + 4 = 20$ bytes per message. The dummy messages are necessary to guarantee continuous flow of traffic, whereas the network benefits from a large number of users.

In contrast to MIX networks, DC-networks can offer perfect sender and receiver unobservability using one time pads, offering the most protection.[15] Chaum [17] proposed a ring topology, where every node is receiving the message from its neighbours and to reduce bandwidth combines his local output directly to the message received. For one DC-net communication round, the message has to travel twice through the ring: the first time is used to get local outputs from neighbouring nodes (sending) and the second one for broadcasting the final, global message to all nodes (receiving).

Here every participant needs to generate one output (XOR operation) for every message bit to traverse the network (superposed sending). This makes the overhead proportional to the number of network participants, since each output needs to be delivered to all other participants (using a reliable broadcast operation). This results in $n(n - 1)$ bits to be processed per 1 bit message, with n being the number of network participants. For a high number of participants this gets very expensive, and gets worse depending on packet size and network collisions. To worsen the situation, both networks require self-organisation and benefit from dummy traffic to ensure a continues flow of traffic. In contrast to MIXing, in DC-net, an increase in the number of messages will also cause an increase in collisions. Thus, taking care of collision handling adds to the overhead for DC-net.

Broadcasts via wireless networks

Wireless networks, however, have one big advantage in terms of topology: broadcasts can be done naturally over the physical medium and therefore are cheaper in terms of communication overhead. This assumes that a broadcast message reaches all other nodes without further interaction such as forwarding. The decoupling between the global message and the clients (as proposed by *Dissent* [20,96]), cannot be realized in a purely decentralized, fully connected network graph. For example if a single node does not broadcast his local message in a given round, the local output cannot be calculated. This is because all other nodes in the network depend on it to calculate the global sum.

[15] Note: computationally secure with public-key encryption.

Comparing costs for different topologies

To compare the costs for the different topologies, the calculations are done twofold: the costs for individual nodes in the DC-net and the overall messages exchanged and their paths in a communication round throughout the whole network. The first aspect is important to determine the load a constrained node has to cope with. The second aspect indicates the bandwidth and latency requirements that a low-power and lossy network like a wireless sensor network has to offer.

Assuming a DC-net with n nodes, with a fully connected key graph, the calculations are done over n rounds of communication. In the ring topology, every node receives the temporary global sum from its predecessor, combines its output and forwards this message. After the temporary global sum has travelled once through the ring and every participant has committed his part, the global sum has to traverse the ring again to be received by every node. The number of sending and receiving steps after n rounds is therefore $2n$ for each node. In the star topology, for $(n-1)$ rounds, one node has only to send his local message and receive the final global sum. But in the round where it has to form the centre, it has to receive $n-1$ and also send $n-1$ messages which sums up to $2n-2$ sending and receiving steps for each node. In the fully connected wireless network, every node has a constant sending and receiving rate, yielding n sending steps and $n \cdot (n-1) = n^2 - n$ receiving steps in total.

Assuming there is only one communication round happening in the ring, only one message may traverse the ring. The load on the ring is thus constant, however, with an increasing number of nodes, the number of hops is increasing linearly and so does the latency. In the star topology, every client sends his message to the star and the star sends the accumulated message back again yielding in $(n-1)^2$ total messages exchanged in one round with quadratic costs. For the fully connected wireless network, exactly n messages are exchanged in total within one round.

Summarizing the costs for individual clients for n rounds, the star topology seems to be favourable. However with the downside that every single node has to cope with one round to build the centre of the star. Nodes in the fully connected network have a constant sending rate of one message per round, but must process $n-1$ per round. The receiving costs are thus increasing linearly with the number of clients. The total number of messages exchanged in the fully connected network and the ring are linear, whereas the costs are quadratic in the star.

4.6 Attacker model

We discussed the different attacker models considered in Sect. 3.4. The attacker model in our use case scenario is a global attacker, who is computationally bound, but can listen to every communication and can insert arbitrary messages, which is very realistic in a locally limited, wireless scenario. However, denial of service attacks, such as through radio jamming, are not considered in this article. Given an attacker with powerful enough radio equipment, which is jamming radio frequencies used by the sensor nodes, there is very little chance of preventing this kind of attack.

Dissent considers these types of attacks by defining thresholds of a minimum number of nodes who have to participate in a given round; otherwise the round is aborted by the servers (or the receiving windows is increased). Unfortunately, this is much more challenging to implement in a decentralized DC-net, because each node participating in a given communication round does not know beforehand, how many nodes will (or can) participate. Therefore, if an attacker is able to simultaneously prevent all other nodes from sending—excluding the victim node—it is possible to isolate it by effectively reducing its anonymity set to the size of "1".

Since a wireless sensor network like in a Smart Home is bound geographically to a specific location, special care has also to be taken to Sybil attacks: an attacker who is physically near to the WSN could—if no counter mechanisms are implemented—flood the network with new "fake-node identities". This can be addressed by limiting the rate at which new nodes can enter the network with the challenge-based protocols which increase the cost to join the network, like proposed in [36].

5 Building unobservable communication using existing sensors with 6LoWPANs and Contiki

In the following, we describe the underlying state-of-the-art technologies for embedded devices. We present all the security mechanisms needed to reduce information leakage and aspire towards unobservable communication, and finally present our feasibility study. In [21,85], we document two higher layer views of the security and privacy issues to be addressed in the IoT in a SmartCity context, which we also take into consideration.

All essential building blocks are available open source [7,53] and we proved them portable to the Re-Mote [6,21,62,64]. Missing system modules were implemented by us in the EU-funded project RERUM[16] [2,70] by [6,62,69]. Within RERUM the performance gains of ECC signatures and DTLS were assessed, both qualitatively and quantitatively, to identify potential issues in the software and hardware modules of the Re-Mote [64].

[16] https://ict-rerum.eu (last acc. 09 January 2019).

5.1 Used sensors and setup

We use the IEEE 802.15.4 for low-power wireless communications [42]. This standard is a key building block of most wireless sensor network (WSN) deployments. It implies that all embedded devices/sensors and at least one gateway must be equipped with an 802.15.4 low-power radio interface. In addition, the ongoing research and standardization efforts in this area have resulted in the adoption of protocols of the TCP/IP family for networks of severely constrained devices. IETF IPv6 over low-power wireless personal area networks (6LoWPAN) [61] and related specifications [15,90] made it possible to use IPv6 in networks of embedded smart objects. For those networks, the IPv6 routing protocol for low-power and lossy networks (RPL) [13] is the de- acto standard for routing, whereas the constrained application protocol (CoAP) [82] is the IETF's protocol recommendation to realise the RESTful [34] architecture for constrained environments.

We use 6LoWPAN [61], RPL [13] and CoAP [82] in our studies. This selection implies that devices are capable of handling the firmware with all above-mentioned stacks and protocols. A suitable firmware for the constrained devices discussed in our use case description in Sect. 3 is the Contiki open source embedded operating system (OS)[17] [18,28].

Contiki is a lightweight operating system designed with IoT and the restrictions and needs of constrained devices in mind. It features a standard-compliant embedded TCP/IP implementation and supports a number of specifications aiming to optimize the use of TCP/IP networking in embedded devices. This includes support for the aforementioned standards and specifications. Contiki officially supports numerous available wireless sensor platforms, including our target platform, the Re-Mote[18] [98] device.

The Re-Mote houses a CC2538 System-on-Chip (SoC) from Texas Instruments. The SoC is based on an ARM Cortex-M3 core which runs at the clock rate of 32 MHz and is supported by 512 kb of programmable flash, 32 kb of SRAM and 4 kb of ROM.[19] For RF communication, the SoC uses a integrated 2.4-GHz 802.15.4 compliant RF transceiver. In addition, to support more computational demanding cryptography operations the CC2538 is capable of executing primitives in hardware. This includes AES-128/256, SHA-2, RSA and as well certain ECC operations. The platform can run Contiki with the needed 6LoWPAN/RPL/CoAP software stacks and protocols.

We follow the RERUM network architecture [2,21,70] thus the gateway provides access to the outside network (e.g., Internet). Sensors will transmit their messages using CoAP over IPv6.

In order for our use case to be implemented using Contiki and those networking technologies mentioned just now, the following needs to be fulfilled:

- All embedded devices and sensors must be equipped with an IEEE 802.15.4 low-power radio interface.
- All embedded devices need to run Contiki, with IETF 6LoWPAN/RPL networking enabled.
- The gateway (Fig. 1) will also get a 802.15.4 interface.
- The gateway creates a 6LoWPAN/RPL network and advertises its presence over the 802.15.4 interface.
- Embedded devices will join this network and, as a result, an 802.15.4 wireless mesh will be formed among those devices and the gateway.
- Sensors use this wireless mesh to transmit their measurements using CoAP over IPv6.

5.2 Security building blocks

In this section, we present an overview of some of the security mechanisms applicable to the smart home networks outlined in the previous section. This includes a discussion of physical security, hop-by-hop security between neighbouring devices, security of routing control messages, end-to-end security, ensuring authenticity and integrity with signatures, and privacy-enhancing technologies using overlay networks.

Physical layer considerations

The original 802.15.4 standard specifies that wireless meshes will use the 2.4 GHz frequency band at a 250 kbps bit rate. However, the 802.15.4g amendment [43] defines alternative physical layers and provisions for wireless operation in different frequency bands, such as at the 863–870 MHz band for Europe. This means that operating at a lower frequency, two devices can now communicate at much greater distances (magnitude of a few kilometres). The amendment also defines various bit rates, ranging from 2.4 to 500 kbps and as well longer frame sizes. The extended distance, if not bounded by devices, means a greater risk of eavesdropping and interception by an external adversary. But then, new rates and longer frame sizes also give an extra flexibility to build networks with the required size and properties.

Hop-by-hop security

The aforementioned 802.15.4 standard specifies security services, which aim to protect the communication between wireless devices on TCP/IP Model Layer 2. To that end, the standard specifies that all the security services use the advanced encryption standard (AES) algorithm with 128-bit

[17] https://contiki-os.org (last acc. 09 January 2019).

[18] https://zolertia.io/product/re-mote-professional-pack/ (last acc. 09 January 2019).

[19] https://ti.com/product/cc2538 (last acc. 09 January 2019).

keys. The standard permits group keys, i.e., a common key used by a group of nodes (devices) mainly for multi-casting and broadcasting. As such a shared group, the key provides protection against outsider nodes, but not against malicious insider nodes sharing the same key.

The specification does not provide details on key generation or distribution, but mentions that keys are provided by higher layers and stored securely. The specification also does not discuss what kind of authentication policies can be applied. The standard defines eight different security suites, which can be used to provide various combinations of confidentiality, integrity and origin authentication. Securing communication on Layer 2 comes with significant performance cost, unless the key is shared by the group. Because then each intermediate node has to perform re-encryption of every single frame using the next-hop node key.

Security of routing control messages

For 6LoWPANs, the de facto standard routing protocol is RPL. It is a distance vector protocol, which perceives the 6LoWPAN network as a tree-like structure called a destination-oriented directed acyclic graph (DODAG). The DODAG is created based on a combination of metrics and constraints known as objective functions and which are used to calculate the best path between a source and destination. The graph building process is initiated at an administratively configured node, which is essentially the tree's root and is often referred to as a border router. Typically, this node is the router connecting the wireless mesh to the Internet, as discussed earlier.

Data traffic in an RPL network can flow upwards in the tree (from a node towards the root), while support for downward flow of data traffic is optional. For point-to-point communication (from any node to any node), datagrams first travel upwards until they reach a node which is a common ancestor to both the source and destination. They are then forwarded downwards to their destination.

For the protection of routing control messages, RPL uses AES-128 CCM as its underlying cryptography algorithm and MAC values can be either 32- or 64-bit long. The RPL specification also discusses support for signed messages, using a scheme based on RSASSA-PSS [48] with 2048- or 3072-bit moduli. Mechanisms defined as part of the RPL specification can only be used for the protection of RPL control packets, but not for application data.

End-to-end security with DTLS

One option to ensure the end-to-end security in IoT is to use the DTLS protocol [54,57]. DTLS is the modified version of a well-known and widely deployed cryptographic protocol called transport layer security (TLS). In contrast to TLS, which requires a reliable TCP channel to establish and carry on a secure communication, DTLS can be used over unreliable protocols, such as the user datagram protocol (UDP). The latter feature, i.e., a possibility of deployment over UDP, makes DTLS a good candidate for secure communications between resource-constrained devices. Furthermore, since CoAP is designed to operate over UDP, it is possible to deploy CoAP over DTLS.

More generally, DTLS is a transport layer protocol that provides authentication of communicating parties together with message integrity and confidentiality. One can distinguish two layers in DTLS, namely the handshake protocol and the record protocol. The first (which itself consists of three sub-protocols) aims for the authentication of communication parties, the negotiation of the cipher suite used, and for a key exchange. On the other hand, the record protocol provides encapsulation capabilities for application-layer protocols.

One of the main tasks of the handshake protocol is to exchange keys that are further used to protect data during communication. DTLS might use both public- and private-key options. Furthermore, the public-key option can use raw public keys as well as full X.509 certificates. The selected configuration is usually a system design choice and since all three options might vary significantly in terms of storage, computation time and power, the advantages and disadvantages of all have to be carefully studied (especially for constrained devices). Once keys are exchanged and parties authenticated, DTLS uses symmetric schemes to provide message integrity and confidentiality.

The above-mentioned pre-placed key option fits well in scenarios where communication parties are able to exchange the symmetric key (using a trusted channel) prior to communication. For instance, the key can be supplied by the manufacturer together with the device firmware. Such a key is stored as a master secret and the session key is derived from it. This pre-placed keys scenario is resource limited and device friendly in terms of computational power. There is no need of public-key operation since only symmetric schemes need to be used. A trade-off is a significant weakened security, since it cannot be guaranteed that keys will not leak (e.g., legal requirement) or cannot be derived (e.g., security vulnerability) at some point in time.

On the other hand, key management is likely to be very costly and inefficient. To mitigate key management efficiency, DTLS allows for two public-key options, namely raw public keys and X.509 certificates. In the former case, the keys are stored in the raw form (without full certificate overheads), which makes storage requirements less demanding. In the latter variant, the full X.509 certificates are used, with all storage and processing time implications.

We developed a research prototype for Contiki and the Re-Mote platform. Our devices are enabled to establish integrity

and confidentiality at the transport layer level, including origin authentication [64,80].

Authentication and integrity protection

The use of per device public keys for confidentiality protection by encryption allows the sender to encrypt data using the intended recipient's public key. Thus, the only recipient capable of deciphering the data is the intended one, assuming the encryption cannot be broken and the private key of the recipient is not leaked. The recipient of such a message has no way of verifying the sender's identity. Digital signatures can be facilitated to provide strong entity authentication. Once the data is signed, the recipient of such a signed message can verify who signed the message and can use the information as a basis for its access control. In Fig. 1, the signed message could include the command 'turn_on', which is now verifiable under the SmartHome Gateway's public signature verification key.

Recently, we presented a proof-of-concept implementation of ECDSA digital signatures using MicroECC [53] for Contiki on the Re-Mote platform for data integrity and data origin authentication (which supports non-reputation) [6,62]. Our prototypical implementation on the Re-Mote device is based on NIST curve P160 ECDSA signatures of JSON encoded sensor data.[20] This allows protecting the integrity of data flowing from, to or between constrained devices. It furthermore allows identifying the origin of data by means of public keys. This works on any application level data, allowing a broad use. The protocol and concept of on device signatures designs are flexible and allow the use of different cryptographic signature mechanisms [6]. It can also be combined with JSON in JSON Sensor Signatures (JSS) format as shown in [69,72].

An example of an JSS message without optimisations is shown in Fig. 4. The figure shows that JSS enables a client to detect modifications of a message.

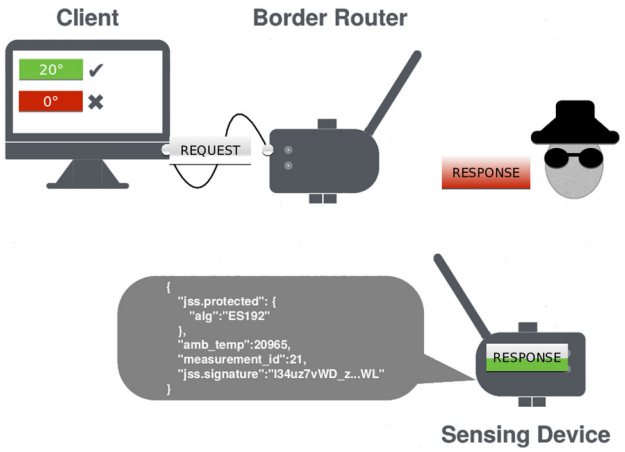

Fig. 4 Example of a JSON sensor signatures (JSS) message (see [69,72] for more details on JSS)

Dining Cryptographers Net (DC-net)

Based on 6LoWPAN and RPL, it is possible to create different overlay networks such as VPN and proxy networks, or aforementioned DC-net/Mixing networks. Security considerations in such networks strongly depend on the number of communicating nodes. We consider DC-net [17] as a preferred local option, and using Mix networks [16] over the Internet.

Chaum [17] proposed a ring topology for DC-net, which was extended by [94], where every node receives the message from its neighbours and to reduce bandwidth adds its local output directly to the message received. For one DC-net communication round, the message has to travel twice through the ring: the first time is used to get local outputs from neighbouring nodes (sending) and the second one for broadcasting the final, global message to all nodes (receiving). The proposed ring topology was implemented by Herbivore [36].

The more recent DC-net implementation 'Dissent' uses a client–server architecture [20,96], where communication always happens among client–server and server–server, but never among two clients directly. Compared to a ring topology, where the slowest client determines the latency and bandwidth characteristics, Dissent increases the performance and scalability (the consequence for the anonymity through this practice are considered through thresholds, i.e., for a communication round, a minimum number of clients have to participate. Wireless networks, however, have one big advantage in terms of topology: broadcasts can be done naturally over the physical medium and therefore are cheaper in terms of communication overhead. This assumes that a broadcast message reaches all other nodes without further interaction such as forwarding. The decoupling between the global message and the clients (as per Dissent), cannot be realized in a purely decentralized, fully connected network graph. For example if a single node does not broadcast its local message in a given round, the local output cannot be calculated. This is due to all other nodes in the network depending on it to calculate the global sum.

The first aspect is important to determine the load a constrained node has to cope with. The second aspect indicates the bandwidth and latency requirements that a low-power and lossy network like a wireless sensor network has to offer. We recently compared these two approaches and implemented a prototype [5]. Summarizing the costs for individual clients, the star topology proved favourable. However with the downside that every single node has to cope with one round to build

[20] https://github.com/ict-rerum (last acc. 09 January 2019).

the centre of the star. Nodes in the fully connected network have a constant sending rate of one message per round, but must process $n - 1$ messages per round. The receiving costs are thus increasing linearly with number of clients. The total number of messages exchanged in the fully connected network and the ring are linear, whereas the costs are quadratic in the star [5].

6 Implementing unobservable communication for the IoT on real sensors

We now present the security features developed, implemented, and/or ported to the Contiki Operating system running on the Re-Mote. With regard to overlay networks providing unobservability discussed in this article, we bring the DC-net protocol [17] to constrained node devices.

Contiki

As of October 2014, Contiki provides off-the-shelf support for 802.15.4 security services [84]. Contiki's source tree includes a software implementation of the AES algorithm, but it also provides drivers for some AES acceleration hardware, such as the CC2538 used in the Re-Mote platform. Currently, the implementation uses pre-shared keys, but efforts have been made to add support for key establishment mechanisms and group communication schemes [44,51].

An evaluation of the RPL security mechanisms for the Contiki, as described in the standard [13], has been introduced in [65]. The authors presented a trade-off between the security configurations and the RPL performance. They provided a lightweight version of the security configuration, which has insignificant impact on the performance (comparing to unsecured RPL) and a full configuration, where the security features have a significant impact on the performance, but on the other hand (in contrast to the lightweight version), protects against replay-based attacks.

DTLS encryption

In the RERUM project we discussed the use of DTLS in [2,21,70,80]. Based on [7] we build a research prototype for Contiki and the Re-Mote platform.[21]

TinyDTLS [7] is an open source project that implements DTLS v1.2 with a focus on constrained devices. Although TinyDTLS is not an official part of Contiki OS, we successfully integrated and used them together. TinyDLTS has a pre-placed key option, which make public-key operations obsolete. Also a raw public-key option is available, which uses the default cipher suite with a standard NIST P-256 elliptic curve. We investigated TinyDTLS in great detail with the focus on security and energy consumption. The experimental results are documented and discussed in [64].

[21] https://github.com/ict-rerum (last acc. 09 January 2019).

Our provisional results show that in case of pre-placed keys the total time of handshake execution is around 1.3 s, whereas for the raw public-key option is around 137 s. The latter case shows significant time overhead to perform the handshake, and thus lower the overall performance of DTLS.

ECDSA signatures

Regarding digital signatures, several implementations for elliptic curve digital signature algorithm (ECDSA) exist. From the existing sets of cryptographic libraries, we selected those, which were suitable without significant underlying changes for both: (1) running under Contiki and (2) running on the ARM Cortex-M3 core. We investigated TweetNaCl (Curve25519 and Ed25519) [11], Piñol (Secp256r1) [68] and MicroECC (Secp160r1, Secp192r1, Secp224r1, Secp256k1, Secp256r1) [53].

To run the aforementioned cryptographic libraries on the Re-Mote, we ported them to Contiki and adjusted the code whenever necessary [62]. As a container we developed JSON sensor signatures (JSS), a JSON format to transport the ECDSA signature over JavaScript object notation (JSON) data alongside [69]. We specifically designed JSS for running on constrained devices and we implemented it in [6,62].

As NIST's curves have recently been accused of being insecure [9] a new ECC-based signature algorithm named Ed25519 was introduced [10]. The underlying curve used in Ed25519 signature is bi-rationally equivalent to the curve Curve25519 [8] and is currently en route to standardization [58]. Therefore, we as well implemented Ed25519 [49] to prove its usability in Contiki [6,62]. We evaluated the total performance and power consumption loss of sending ECC signed messages, which also tripled the message size, on IoT devices to be about 300% [62].

However, there is still room for optimisation for both the IoT hardware and for Contiki.

DC-net

In the following we present our proof-of-concept implementation. We also integrated some of the improvements suggested in [93,94]. The feasibility of our implementation is discussed in [5].

As mentioned in [36], there should be an entry burden for newly connecting nodes to reduce the risk of Sybil attacks. For attacker nodes who are already in the DC-net this adds a penalty if they get detected and kicked out of the DC-net, since reconnecting (maybe with another identity) is more costly. This is especially important, because in many IoT and WSN environments, such as Smart Cities, attackers cannot be easily located and removed. Each new node that enters the network has to establish secrets with all other nodes. This can be done using the Diffie–Hellman key exchange. Such key exchange based on elliptic curve cryptography making use of

ECDH can be done efficiently with the *MicroECC*[22] library. Note that we did not implement the entry protocol in our proof-of-concept implementation. In the prototype, there are currently only two RE-Mote nodes with hardcoded shared secrets.

In contrast to the original protocol, we implemented important optimisations for the message exchange:

Instead of sending 1 bit messages in every message round, we exchange messages with an increased payload of 7 bytes. This is done by carrying multiple message rounds in only one broadcast packet in the "sending vector". The prototype currently uses four message rounds per packet, which is a rather arbitrary number that needs to be optimized in future. This permits the receiver to spend less effort on putting together the individual messages from the clients. A message round is a single round, where each participant commits the combination of its shared secrets and where one single client sends its communication payload. Another optimisation is that we exchange multiple messages within a single 802.15.4 frame, to utilise the network more efficiently. In total, this significantly decreases effort for the recipient to reassemble the individual messages from the other member in the network.

In the time of writing, our prototype does not utilise a TCP or a UDP stack. Instead, it builds upon the RIME [27] stack, which is a network stack of the Contiki OS [28] with very low overhead. RIME itself offers different layers with certain services, but our prototype uses solely the "anonymous broadcast" layer, which offers the required broadcast service. This RIME layer builds directly upon 802.15.4, which permits a node to send 127 bytes at most. Therefore, it increases the efficiency in terms of energy consumption and overhead by utilising the packets as well as possible. Instead of transporting only one message round per packet (i.e., 1 bit), multiple message rounds are carried in only one broadcast packet in the "sending vector". The prototype currently uses four message rounds per packet, but this is a rather arbitrary number that needs to be optimized in future.

In every message round, every participant commits his local message by combining its shared secrets with the payload. But in contrast to the initial DC-net protocol with single bits and the XOR operation, an Abelian group G modulo a prime p is used, improvements by Waidner and Pfitzmann [93,94]. For this first proof of concept, we decided to choose 8 bytes for the message size. The biggest prime that can be expressed in 8 bytes is 18, 446, 744, 073, 709, 551, 557. Rounded down to 7 bytes, every message payload with 7 bytes can be encoded as a number n with $n \in G(\mathbb{N}/18, 446, 744, 073, 709, 551, 557\mathbb{N}, +)$. An Abelian group modulo a prime was chosen as it allows to easily calculate the inverse of a number $n : n^{-1} = p - n$.

[22] https://kmackay.ca/micro-ecc (last acc. 09 January 2019).

Now, considering two participants A and B with the shared secret $s \in G(\mathrm{N}/p\mathrm{N}, +)$ and A actively sending the number $n \in G$, the global sum is calculated as follows:

$$\begin{aligned}
&(\mathrm{Local}_A + \mathrm{Local}_B) \mod p \\
&= ((s + n) + (s^{-1} + 0)) \mod p \\
&= (s + s^{-1} + 0 + n) \mod p \\
&= n
\end{aligned} \tag{3}$$

The reservation is done with the reservation map technique described in [94], based on superposed sending using the same Abelian group previously described. Each client who wants to send messages randomly chooses positions in the reservation vector and indicates this by setting the specific positions to 1. All other positions are set to 0. Nodes that do not want to send messages send an empty reservation vector with zeros in all positions. Afterwards, every node applies its shared secrets to all slots. Each client broadcasts this vector through the network and receives the reservation vectors from all other clients. After every client has received all reservation vectors, it sums the vectors component together. Afterwards, positions in the vector with value 0 mean: no reservation, positions with value 1 mean: successful reservation of exactly one node and positions with value > 1 indicate a collision. The reservation vector consists of a single magic byte at the beginning indicating that this is a reservation vector and four 8 byte slots containing the reservation number.

In the transmission phase, every node prepares its vectors: 0s and the applied shared secrets in the passive sending slots and n and the applied shared secrets for the active sending slot (if any slot was reserved successfully) with n being the payload encoded as a group number. The transmission vector is of equal size as the reservation vector with a different magic byte at the beginning. The nodes send their transmission vector and receive all vectors from the others, combine the vectors and are finally able to see the payloads in each slot for each message round.

If a node wants to leave the network, it is important to have signalling mechanisms so that other nodes are informed and the computation of the global sum does not rely on the local result from the leaving node. Such an exit phase is currently not implemented in our prototype.

7 Complementary privacy-enhancing mechanisms

Privacy can be attacked and protected in many different ways. The protection of the local network communication between trusted devices is too often left out of the picture. In this article, we try to prevent an local attacker to learn by eavesdropping on network traffic which is a very powerful generic

attack vector. Nevertheless, our approach does not protect against the data-receiving back-end collecting potentially unnecessary data from the communicating devices. In this complementary section, we briefly discuss additional layers of protection and the necessary architectural enablers.

7.1 Minimizing data collection

In the early phase of IoT development, there was a trend that next to no smartness was running in a constrained device near to the user. This negatively impacts user privacy as it sends a lot of data to servers elsewhere. Furthermore to comply with EU data protection regulations, it requires an informed consent. Technically it requires an implicit trust in the Internet and all intermediate network hops as well as the service. This trust has not been proven justified yet.

While end-to-end security with privacy-enhancing communication techniques helps to protect data in transit, it does not protect the data once the intended recipient received all the information. Hence, "the best protection for personal data is to not have collected the data at all. Data minimization can be seen as an example of the principle 'prevention is better than reaction'." [33]. In this context "prevention" means that one tries to avoid storing personal data as much as possible, thereby reducing the need for "reaction", i.e., securing stored data. This is known as the principle of data minimisation and can be found in European legal texts, like the GDPR [32], or technical guidelines on privacy like ISO 29100 [45].

In detail, the Article 5 GDPR on "Principles relating to processing of personal data" describes the data minimisation principle as follows:

> "1. Personal data shall be: [...] adequate, relevant and limited to what is necessary in relation to the purposes for which they are processed ('data minimisation');" [Art. 5 (c) GDPR]

In a first step each IoT device should minimize the data being collected. This requires little computing power on the device itself, but rather a smart configuration which tailors the data collection in terms of granularity of the data towards each application. Minimization can occur simply by not collecting data as frequently as technically possible or by sensing not too accurately which results in a resolution reduction either in the axis of time or precision. A continuative step is to process the data before it is being send, i.e., by aggregation or perturbation of the actually collected data.

While data minimization causes a loss in the data quality, it might be in many cases still acceptable for the intended application. As a case study on real-life energy consumption data notes: "simple presence detection is still feasible on the processed data set, more detailed inferences requiring higher temporal or energy-level details are clearly aggravated." [73]. Thus, a loss of data quality occurring could be a privacy gain whenever it does not harm the intended functionality in the back-end. However, too fine-grained values have been shown to allow deductions that might be too invasive. For an example, take the energy metre that used to be able to detect how much electrical energy was used by a household for billing purposes on a monthly basis. If such a metre sends too fine-grained energy values, the resulting data collected by the recipient of the data allows for detecting appliances within the household [59], detecting the use mode of the appliances [30] as well as deducting the residential customers' behaviour [52].

To only send the information required for a specific service, it needs some local smartness to carry out the necessary local pre-processing of the collected sensory data. While one might argue that it is too much for a smaller IoT device, most of the gateways offer sufficient computational resources, which can be controlled and used. This is sometimes also known as fog or edge computing, because it moves intelligence and processing power towards the network's edge rather than having it centrally, e.g., in a server. Once computational resources are locally available, they can be used to process data before sharing outside the local network and to apply additional privacy preserving techniques. The choice of suitable privacy-enhancing processing mechanisms strongly depends on the data the device is collecting (for energy consumption see [46] and for location information see [22,78]).

To conclude, any attempt to apply data minimisation obviously reduces the data's quality compared to the technically possible maximum. However, it thus hinders any non-intended and privacy-invasive application. Of course, care must be taken to apply it such that it is preserving the utility of the data.

7.2 Attribute-based encryption

In some circumstances, however, data minimization techniques might be less applicable and, i.e., to prevent the reduction of data quality by the above-mentioned technique, solutions based on attribute-based encryption (ABE) [38] might be also considered. In that case, under the assumption that data that is being kept on the server is not a subject of elaboration, data might be stored in the encrypted form. Under an ABE scheme, encrypted data are labelled with a set of attributes and, in that setup, a decryption is only possible using a key which attributes matches the encrypted data attributes. Applications of ABE in the IoT context were presented in, i.e., [75,83] where the authors introduced the ABE cities, an encryption system for urban sensing that allows for a fine-grained access control over the encrypted data or augmented a well-known in IoT space MQTT and MQTT-SN protocols with ABE. Since ABE encryption techniques are based on bilinear pairing operations (a set of operations that are rather difficult to implement in a lightweight fash-

ion in constrained devices), a novel no pairing ECC-based lightweight attribute-based encryption scheme was introduced in [97] to address this issue.

7.3 Mediated device access

A more powerful gateway can act as a privacy guardian for less powerful IoT devices at the edge. In this architecture, all communication between the sensors and/or actuators and a server are mediated via a gateway, not allowing any direct uninterceptable communication between devices. While the gateway cannot make those devices communicate unobservable amongst themselves, it can do the data minimisation and pre-processing before information flows towards outside servers. This is known also under the term of mediated device access and indeed core to many IoT architectures, also those from RERUM and SEMIOTICS.[23]

To conclude, an architecture that channels all data through gateways and then harvests the gateway's smartness, i.e., its local processing power and local knowledge of the local deployment, becomes an enabler for privacy. Among many things, it allows to preserve privacy by decoupling local detection–reaction loops that are otherwise are observable by third parties from the outside from internal. Note, that although this seems to oppose any cloud or server-based solutions, it can also complement such solutions as it allows to utilise potentially more expensive or sparse network resources efficiently. Last but not least, the gateway offers to harmonise many different networking and communication technologies [63] such it eases upstream communication.

7.4 Authenticity preserving but privacy-enhancing processing

All the pre-processing of the sensed data for privacy enhancements is done distributed, e.g., not at the data sink. It might be done already at the data source, but most often the architecture foresees it to happen in between the data collection point and the point of the actual data user. This raises a conflict to preserve the authenticity, i.e., the integrity and the origin of the data. Following Sect. 5.2, it is technically feasible to digitally sign the collected data on the sensor and that the application processing the data is verifying those signatures [6,62,64,72]. Then any pre-processing would mitigate privacy issues raised by overly precise values, e.g., by adding noise or another data perturbation mechanisms. However, changing the data would also invalidate any regular digital signature.

The cryptographic methods of malleable signatures could be of help [12], since they allow the signer to pre-define allowed subsequent changes to the signed data. If the subsequent changes stay within those authorised changes, then the signature is still valid and the recipient can verify that no unauthorised changes have happened, thus verifies a reduced integrity, but a much higher one compared to the failed classical signature. For example, this cryptographic tool was used in [71] to set the limit of perturbation on energy consumption values from a Smart Metering Gateway that could be added by a privacy gateway. It also allowed the privacy gateway to inform the user of the allowed changes, i.e., alert if the allowed noise level is too low and thus too fine-grained data is requested.

To conclude, the protection of integrity and the authentication of origin of data coming from the IoT devices can be protected by cryptography even in the light of foreseen subsequent modifications by facilitating malleable signatures [12], e.g., redactable signature schemes [47,88].

7.5 Architectural enablers to improve privacy

The IoT network architecture defines its flexibility to accommodate missing functionality to improve privacy and security. For example, the reference architecture researched by the EU lighthouse project IoT-A[24] was further evolved by the authors of this work within the EU research project RERUM.[25]

RERUM greatly enhanced the IoT-A architecture and catered for encrypted communication beyond that of hop-to-hop communication. This includes end-to-end security with DTLS and integrity protection with ECDSA-based signatures applied on JSON sensor data [6,62,72] as mentioned in Sect. 5.2. Both of the above functionality shows that the ever-increasing local processing power of smart IoT devices can be put to a secure use. While the unobservable communication requires more steps, it could not work without an architecture that treats devices as individual entities and foresees that they have their own security key material.

7.6 Flexible network architectures

In the architecture of the research project SEMIOTICS, the complete network between the IoT gateway and the applications is based on software-defined networking (SDN). SDN decouples the network control from the data forwarding plane. This allows to have more flexible network functionality and also to implement security functionality [1,74]. While there are new privacy problems in SDN, e.g., by the need to share information about local network topologies to the applications configuring or optimizing them [1] the logically separated control plane allows to implement

[23] https://semiotics-project.eu (last acc. 09 January 2019).

[24] https://iot-a.eu (last acc. 16 November 2016 (now offline)).

[25] https://ict-rerum.eu (last acc. 09 January 2019).

more than just encrypted links or separation of network segments. In SDN, "new control functions can be implemented by writing software-based logic in the control plane which deploys the decision logic in the forwarding plane through standard interfaces" [1]. This also allows to implement functionality that is "[a]kin to onion routing" [55]. The authors of [55] use the SDN to implement logic that rewrites packet headers to increase the privacy by hiding the actual flow of messages an observer on the network. The authors of [29] propose to "move onion-routing technique to the SDN" and it "builds onion-routed tunnels over multiple anonymity service providers and through many SDNs" [29].

Thus, a flexible network—and SDNs are designed to be very flexible—will ease to change or adapt the networks core routing to foster anonymous communication. If used between the gateway and the applications, it helps to provide unobservable communication beyond the gateway and thus is an enabler to reach full privacy in the IoT starting from the enabling UC for the communication between the IoT devices itself as described in this article.

8 Conclusions

Privacy cannot be retrofitted, and for privacy this rule holds even more than in security. Thus, if we want to support sensitive services in the Internet of Things, we must act now to put enabling technology in place, and it is best placed into each IoT device. In this article we have shown that we have already implemented most of the building blocks required for secure and private communication in the communication realm of the IoT devices itself: this includes DTLS for encryption, ECDSA signatures for integrity and authentication of origin and DC-net for unobservable communications.

In particular, our proof-of-concept of DC-net on the Re-Mote shows that this technique can indeed be brought to the level of device to device communications of the Internet of Things. This contribution highlights that the Internet of Things does not need to fall behind in comparison to the privacy achievable on the Internet. Of course, many parameters, like the optimal message size, an efficient key exchange or the influence of disturbers have to be further considered. However, our proof-of-concept shows that strong, privacy enhancing technologies can be adapted to use in the IoT. The implementation currently relies on reliable broadcast assumption, which might not hold in real world use cases considering bigger networks like Smart Cities.

We have further discussed enabling technology in the area of IoT that allows to foster privacy, i.e., flexible networking technology like SDNs could implement privacy preserving routing overlays, local data pre-processing on the IoT gateway could minimize the data being sent to servers, while advanced cryptographic mechanisms of malleable signatures still guarantee the absence of unauthorised subsequent changes to the application. All these technological advances brought to the IoT and integrated into each device should allow it to become a safe place—one where we can accept such otherwise too privacy-invasive technology into our daily lives, or into our restricted corporate areas if we consider industrial Internet of Things (IoT) where the threat of privacy is a threat of trade secret protection and corporate espionage.

9 Recommendations

We learned from our prototype implementation of DC-net on real hardware that to take the final steps towards a truly private communication between IoT devices, the following recommendations shall be followed:

Minimize data collection Data you do not need to fulfil a goal shall not be collected by the IoT.

Increase the ROM footprint of devices This is a call to hardware manufacturers, its of course not to store even more data on the devices, but to allow all security mechanisms to fit to make the IoT device a first-class citizen in secure and private communication. All the security mechanisms discussed in Sect. 5 should be in place, or allow over the air provisioning. But even with the latter combining, all of them in the same firmware results in a binary image too large and therefore some needed security mechanisms may get excluded in a real deployment.

Support hardware acceleration This second recommendation towards hardware has the obvious benefit that encryption and decryption operations are potentially faster when accelerated by hardware. But it can also reduce the size of the firmware image. However, besides the runtime, other aspects like the problem of side channel attacks have to be considered in greater depth.

Enable layer 2 hop-by-hop encryption within the wireless network: This step prohibits the attacker from reconstructing the network's topology using a very simple wireless sniffer on the RPL routing control packets transmitted in the clear.

Carefully consider the use of devices that allow long range communication, e.g., sub-GHz frequency band It has the disadvantage that the attacker does not need to be so close to eavesdrop. On the other hand, it will allow to broadcast messages to a wider audience, which can significantly reduce the overhead of the DC-net protocol.

Optimize network algorithms of lower layers To minimize the negative impact that DC-net's communication overhead generates the lower network protocols could be further optimized to better cope with constant traffic, e.g., adapt solutions from constant-rate transmissions like streaming.

As part of our future work, we continue to optimize the implementation discussed in Sect. 4 and provide further mea-

surements on their impact on network delay, packet loss and device energy consumption.

Acknowledgements We would like to thank J. Bauer, G. Oikonomou, B. Petschkuhn, M. Moessinger, M. Danish, and A. Moore for their invaluable help during the course of this work. H. C. Pöhls and R. C. Staudemeyer were partially funded by the European Union's FP7 grant no. 609094 (RERUM) and the EU's H2020 grant no. 644962 (PRISMACLOUD). H. C. Pöhls was also funded by the EU's H2020 grant no. 780315 (SEMIOTICS). M. Wójcik was partially funded by the EU's H2020 grant no. 644866 (SSICLOPS). This paper reflects only the authors' views.

References

1. Ahmad I, Namal S, Ylianttila M, Gurtov A (2015) Security in software defined networks: a survey. IEEE Commun Surv Tutor 17(4):2317–2346. https://doi.org/10.1109/COMST.2015.2474118
2. Angelakis V, Cuellar J, Fischer K, Fowler S, Gessner J, Gundlegård D, Helgesson D, Konios G, Lioumpas A, Lunggren M, Mardiak M, Moldovan G, Mouroutis T, Nechifor S, Oikonomou G, Pöhls HC, Ruiz D, Siris V, Suppan S, Stamatakis G, Stylianou Y, Traganitis A, Tragos EZ (2014) The RERUM system architecture. Tech. rep., University of Passau
3. Baldini G, Peirce T, Botterman M, Talacchini MC, Pereira A, Handte M, Rotondi D, Pöhls HC, Vermesan O, Baddii A, Copigneaux B, Schreckling D, Vigano L, Steri G, Piccione S, Vlacheas P, Stavroulaki V, Kelaidonis, D., Neisse, R., Tragos E, Smadja P, Hennebert C, Serrano M, Severi S, Abreu G, Kirstein PT, Varakliotis S, Skarmeta A (2015) Internet of Things: IoT governance, privacy and security issues. In: Position paper activity chain 05, IERC-European research cluster on the Internet of Things
4. Bandyopadhyay D, Sen J (2011) Internet of things: applications and challenges in technology and standardization. Wirel Pers Commun 58(1):49–69
5. Bauer J, Staudemeyer RC (2017) From dining cryptographers to dining things: unobservable communication in the IoT in practice. In: Proceedings of the international workshop on computer-aided modeling analysis and design of communication links and networks (CAMAD'17), p 9
6. Bauer J, Staudemeyer RC, Pöhls HC, Fragkiadakis A (2016) ECDSA on things: IoT integrity protection in practice. In: Proceedings of the 18th international conference on information and communications security (ICICS'16). Springer, pp 1–15. Retrieved from https://projects.eclipse.org/projects/iot.tinydtls. Accessed 29 June 2016
7. Bergmann O (2015) TinyDTLS: a DTLS open source stack. Retrieved from https://projects.eclipse.org/projects/iot.tinydtls. Accessed 29 June 2016
8. Bernstein DJ (2006) Curve25519: New Diffie–Hellman speed records. In: Proceedings of the int. workshop on public key cryptography (PKC'06), LNCS, vol 3958. Springer, pp 207–228
9. Bernstein DJ, Chou T, Chuengsatiansup C, Hülsing A, Lambooij E, Lange T, Niederhagen R, van Vredendaal C (2014) How to manipulate curve standards: a white paper for the Black Hat. Secur Stand Res LNCS 9497:109–139
10. Bernstein DJ, Duif N, Lange T, Schwabe P, Yang By (2012) High-speed high-security signatures. J Cryptogr Eng 2(2):77–89
11. Bernstein DJ, van Gastel B, Janssen W, Lange T, Schwabe P, Smetsers S (2014) TweetNaCl: a crypto library in 100 tweets. In: Proceedings of the international conference on cryptology and information security in Latin America (LATINCRYPT'14), vol 8895
12. Bilzhause A, Pöhls HC, Samelin K (2017) Position paper: the past, present, and future of sanitizable and redactable signatures. In: Proceedings of international conference on availability, reliability and security (ARES 2017), pp 87:1–87:9. ACM. https://doi.org/10.1145/3098954.3104058. Sep 2017
13. Brandt A, Hui J, Kelsey R, Levis P, Pister K, Struik R, Alexander R (2012) RFC6550—RPL: IPv6 routing protocol for low-power and lossy networks, Winter T, Thubert P (eds). https://doi.org/10.17487/rfc6550
14. Cavoukian A (2009) 7 Foundational Principles - Privacy By Design. Retrieved from https://www.privacybydesign.ca/index.php/about-pbd/7-foundational-principles. Accessed 27 July 2015
15. Chakrabarti S, Nordmark E, Bormann C (2012) RFC6775—neighbor discovery optimization for ipv6 over low-power wireless personal area networks (6LoWPANs) Shelby Z (ed). https://doi.org/10.17487/rfc6775
16. Chaum DL (1981) Untraceable electronic mail, return addresses, and digital pseudonyms. Commun. ACM 24(2):84–90
17. Chaum DL (1988) The dining cryptographers problem: unconditional sender and recipient untraceability. J Cryptol 1(1):65–75
18. Contiki (2017) Contiki: the open source OS for the internet of things. Retrieved from http://www.contiki-os.org/. Accessed 27 May 2017
19. Cooper DA, Birman KP (1995) The design and implementation of a private message service for mobile computers. Wirel Netw 1(3):297–309
20. Corrigan-Gibbs H, Ford B (2010) Dissent: accountable anonymous group messaging. In: Proceedings of the 17th ACM conference on computer and communications security (CCS'10), pp 340–350. ACM
21. Cuellar J, Bauer J, Fragkiadakis A, Petschkuhn B, Pöhls HC, Ruiz D, Tragos EZ, Staudemeyer RC, Suppan S, Weber R, Wójcik M (2015) Privacy enhancing techniques in Smart City applications. Tech. rep., University of Passau
22. Cuellar J, Ochoa M, Rios R (2012) Indistinguishable regions in geographic privacy. In: Proceedings of the 27th annual ACM symposium on applied computing, SAC '12. ACM, New York. pp 1463–1469. https://doi.org/10.1145/2245276.2232010
23. Danezis G, Clayton R (2007) Introducing traffic analysis. In: Digital privacy: theory, technologies, and practices, pp 1–24
24. Danezis G, Dingledine R, Mathewson N (2003) Mixminion: design of a type III anonymous remailer protocol. In: Proceedings of the symposium on security and privacy. IEEE, pp 2–15
25. Danezis G, Domingo-Ferrer J, Hansen M, Hoepman JH, Metayer DL, Tirtea R, Schiffner S (2014) Privacy and data protection by design—from policy to engineering. Tech. Rep. dec, European Union Agency for Network and Information Security
26. Dingledine R, Mathewson N, Syverson P (2004) Tor: The second-generation onion router. In: Proceedings of the 13th USENIX security symp., vol 13. USENIX Association. pp 303–320
27. Dunkels A (2007) RIME—a lightweight layered communication stack for sensor networks. In: Proceedings of the European conference on wireless sensor networks (EWSN'07), Poster Abstract, p 2
28. Dunkels A, Grönvall B, Voigt T (2004) Contiki—a lightweight and flexible operating system for tiny networked sensors. In: 29th annual international conference on local computer networks (LCN'04), pp 455–462

29. Elgzil A, Chow CE, Aljaedi A, Alamri N (2017) Cyber anonymity based on software-defined networking and onion routing (sor). In: 2017 IEEE conference on dependable and secure computing, pp 358–365. https://doi.org/10.1109/DESEC.2017.8073856
30. Enev M, Gupta S, Kohno T, Patel SN (2011) Televisions, video privacy, and powerline electromagnetic interference. In: Proceedings of ACM SIGSAC symposium on information, computer and communications security (ASIA CCS 2011), pp 537–550. ACM
31. EU Article 29 Data Protection Working Party (WP 223): Opinion 8/2014 on the recent developments on the Internet of Things (2014)
32. (2016) European Parliament and the Council of the European Union: Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation). Off J 1–88. **(OJ L 119 of 4.5.2016)**
33. European Union Agency for Network and Information Security (ENISA): recommended cryptographic measures-securing personal data (2013). URL https://www.enisa.europa.eu/publications/recommended-cryptographic-measures-securing-personal-data/at_download/fullReport
34. Fielding RT (2000) Architectural styles and the design of network-based software architectures. Ph.D. thesis, University of California, Irvine
35. Frizell S (2014) This startup is trying to create—and control—the Internet of your home. TIMES Magazine
36. Goel S, Robson M, Polte M, Sirer E (2003) Herbivore: a scalable and efficient protocol for anonymous communication. Tech. rep., Cornell University
37. Golle P, Juels A (2004) Dining cryptographers revisited. In: Proceedings of advances in cryptology (EUROCRYPT '04), vol 2729, pp 456–473
38. Goyal V, Pandey O, Sahai A, Waters B (2006) Attribute-based encryption for fine-grained access control of encrypted data. In: Proceedings of the 13th ACM conference on Computer and communications security, pp 89–98. ACM
39. Guan Y, Fu X, Bettati R, Zhao W (2002) An optimal strategy for anonymous communication protocols. In: Proceedings of the 22nd international conference on distributed computing systems (ICDCS'02), pp 257–266. IEEE
40. Hewlett Packard Enterprise (2015) Internet of Things research study. Tech. Rep. jul, HP
41. Hewlett Packard Enterprise (2015) Internet of Things Security Study: home security systems report. Tech. rep., HP
42. IEEE Standards Association: Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs) (2006)
43. IEEE Standards Association: Part 15.4g: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 3: Physical Layer (PHY) Specifications for Low-Data-Rate, Wireless, Smart Metering Utility Networks IEEE Computer (2012)
44. Ilia P, Oikonomou G, Tryfonas T (2013) Cryptographic key exchange in IPv6-based low power, lossy networks. In: Workshop in information security theory and practice (WISTP'13), lecture notes in computer science, vol 7886, pp 34–49. Springer
45. ISO/IEC: ISO/IEC 29100:2011: information technology—security techniques—privacy framework. ISO, Geneva (2011)
46. Jawurek M (2013) Privacy in smart grids. Ph.D. thesis, Friedrich-Alexander-University Erlangen-Nuernberg
47. Johnson R, Molnar D, Song D, Wagner D (2002) Homomorphic signature schemes. In: Proceedings of the RSA security conference—cryptographers track. Springer, pp 244–262
48. Jonsson J, Kaliski B (2003) RFC3447—Public-Key Cryptography Standards (PKCS) #1: RSA cryptography specifications version 2.1. Tech. Rep. 3447, Internet Engineering Task Force
49. Josefsson S, Liusvaara I (2017) Edwards-Curve Digital Signature Algorithm (EdDSA). RFC 8032. 10.17487/RFC8032
50. Kelly D, Raines R, Baldwin R, Grimaila M, Mullins B (2012) Exploring extant and emerging issues in anonymous networks: a taxonomy and survey of protocols and metrics. IEEE Commun Surv Tutor 14(2):579–606
51. Krentz KF, Rafiee H, Meinel C (2013) 6LoWPAN security: adding compromise resilience to the 802.15.4 security sublayer. In: Proceedings of the international workshop on adaptive security (ASPI'13), pp 1–10. ACM
52. Lisovich MA, Mulligan DK, Wicker SB (2010) Inferring personal information from demand-response systems. IEEE Secur Priv 8(1):11–20. https://doi.org/10.1109/MSP.2010.40
53. MacKay K (2016) micro-ecc. Retrieved from http://kmackay.ca/micro-ecc/. Accessed 2 Oct 2016
54. McGrew D, Rescorla E (2010) RFC5764—datagram transport layer security (DTLS) extension to establish keys for the secure real-time transport protocol (SRTP). Tech. rep., RFC Editor
55. Meier R, Gugelmann D, Vanbever L (2017) itap: In-network traffic analysis prevention using software-defined networks. In: Proceedings of the symposium on SDN research, SOSR '17. ACM, New York, pp 102–114. https://doi.org/10.1145/3050220.3050232
56. Miorandi D, Sicari S, De Pellegrini F, Chlamtac I (2012) Internet of things: vision, applications and research challenges. Ad Hoc Netw 10(7):1497–1516
57. Modadugu N, Rescorla E (2004) The design and implementation of datagram TLS. In: Proceedings of the 11th annual network and distributed system security symposium (ISOC NDSS'04)
58. Moeller N, Josefsson S (2015) IETF draft: EdDSA and Ed25519. Retrieved from https://tools.ietf.org/html/draft-josefsson-eddsa-ed25519-02. Accessed 2 Oct 2016
59. Molina-Markham A, Shenoy P, Fu K, Cecchet E, Irwin, D (2010) Private memoirs of a smart meter. In: Proceedings of 2nd ACM BuildSys '10. ACM, pp 61–66. https://doi.org/10.1145/1878431.1878446
60. Möller U, Cottrell L, Palfrader P, Sassaman L (2003) Mixmaster protocol—version 2 Internet Draft, July. Retrieved from https://www.ietf.org/archive/id/draft-sassaman-mixmaster-03.txt
61. Montenegro G, Kushalnagar N, Hui J, Culler, D (2007) RFC4944—transmission of IPv6 packets over IEEE 802.15.4 networks. https://doi.org/10.17487/rfc4944
62. Mössinger M, Petschkuhn B, Bauer J, Staudemeyer RC, Wójcik M, Pöhls HC (2016) Towards quantifying the cost of a secure IoT: overhead and energy consumption of ECC signatures on an ARM-based device. In: 17th international symposium on a world of wireless, mobile and multimedia networks (WoWMoM). IEEE, p 6
63. Palavras E, Fysarakis K, Papaefstathiou I, Askoxylakis I (2018) Semibiot: secure multi-protocol integration bridge for the iot. In: 2018 IEEE international conference on communications (ICC), pp 1–7. https://doi.org/10.1109/ICC.2018.8422486
64. Papadopoulos G, Staudemeyer RC, Wójcik M, Pöhls HC, Oikonomou G, Angelakis V, Bauer J, Charalampidis P, Fragkiadakis A, Gundlegård D, Katuri S, Makrogiannakis A, Petschkuhn B, Stamatakis G, Surligas M, Tragos EZ, Fragkiadakis A, Tragos EZ, Papadopoulos G, Gundlegård D, Angelakis V, Katuri S, Bauer J, Petschkuhn B, Charalampidis P, Stamatakis G, Surligas M, Makrogiannakis A (2016) The RERUM laboratory evaluation results. Tech. rep., University of Passau
65. Perazzo P, Vallati C, Arena A, Anastasi G, Dini G (2017) An implementation and evaluation of the security features of RPL. In: Puliafito A, Bruneo D, Distefano S, Longo F (eds) Ad-hoc, mobile, and wireless networks. Springer International Publishing, Cham, pp 63–76
66. Pfitzmann A, Hansen M (2010) A terminology for talking about privacy by data minimization: anonymity, unlinkability, unde-

tectability, unobservability, pseudonymity, and identity management. Tech. rep., Technical University Dresden
67. Pfitzmann A, Pfitzmann B, Waidner M (1991) ISDN-mixes: untraceable communication with very small bandwidth overhead. In: GI/ITG-conference "Kommunikation in verteilten Systemen" (communication in distributed systems), pp 451–463
68. Piñol Piñol O (2014) Implementation and evaluation of BSD elliptic curve cryptography. Master thesis (pre-bologna period), Universitat Politècnica de Catalunya
69. Pöhls HC (2015) JSON Sensor Signatures (JSS): end-to-end integrity protection from constrained device to IoT application. In: Proceedings of the workshop on extending seamlessly to the Internet of Things (esIoT'15). IEEE, pp 306–312
70. Pöhls HC, Angelakis V, Suppan S, Fischer K, Oikonomou G, Tragos EZ, Rodrigo Diaz Rodriguez, Mouroutis T (2014) RERUM: building a reliable IoT upon privacy- and security-enabled smart objects. In: Wireless communications and networking conference workshop on IoT communications and technologies (WCNC '14), pp 122–127. IEEE
71. Pöhls HC, Karwe M (2014) Redactable signatures to control the maximum noise for differential privacy in the smart grid. In: Proceedings of international workshop on smart grid security (SmartGridSec 2014), LNCS, vol 8448. Springer, pp 79–93. https://doi.org/10.1007/978-3-319-10329-7_6
72. Pöhls HC, Petschkuhn B (2017) Towards compactly encoded signed IoT messages. In: Proceedings of IEEE international workshop on computer-aided modeling analysis and design of communication links and networks (IEEE CAMAD 2017). IEEE, pp 1–6. https://doi.org/10.1109/CAMAD.2017.8031622. http://henrich.poehls.com/papers/2017_PoehlsPetschkuhn_IoT_signature_encoding_CAMAD.pdf. Accessed: Sep 2017
73. Pöhls HC, Petschkuhn B, Rückert J, Mössinger M (2014) Aggregation and perturbation in practice: case-study of privacy, accuracy and performance. In: IEEE international workshop on computer-aided modeling analysis and design of communication links and networks (IEEE CAMAD 2014). IEEE, pp 183–187. https://doi.org/10.1109/CAMAD.2014.7033231
74. Raghavan B, Casado M, Koponen T, Ratnasamy S, Ghodsi A, Shenker S (2012) Software-defined internet architecture: Decoupling architecture from infrastructure. In: Proceedings of the 11th ACM workshop on hot topics in networks, HotNets-XI. ACM, New York, pp 43–48. https://doi.org/10.1145/2390231.2390239
75. Rasori M, Perazzo P, Dini G (2018) ABE-Cities: an attribute-based encryption system for smart cities. In: 2018 IEEE international conference on smart computing (SMARTCOMP). IEEE, pp 65–72
76. Raymond JF (2001) Traffic analysis: protocols, attacks, design issues, and open problems. In: Federrath H (ed) Designing privacy enhancing technologies, LNCS. Springer, pp 10–29
77. Reed MG, Syverson PF, Goldschlag DM (1998) Anonymous connections and onion routing. J Sel Areas Commun 16(4):482–494
78. Rios R, Lopez J, Cuellar J (2016) Location privacy in wireless sensor networks, 1st edn. CRC Press Inc, Boca Raton
79. Roman R, Zhou J, Lopez J (2013) On the features and challenges of security and privacy in distributed internet of things. Comput Netw 57(10):2266–2279
80. Ruiz D, Wójcik M, Pöhls HC et al (2015) Enhancing the autonomous smart objects and the overall system security of IoT based smart cities. Tech. rep., University of Passau
81. Rupprecht D, Kohls K, Holz T, Pöpper C (2019) Breaking LTE on layer two. In: IEEE symposium on security & privacy (SP). IEEE
82. Shelby Z, Hartke K, Bormann C (2014) RFC7252—the constrained application protocol (CoAP)
83. Singh M, Rajan M, Shivraj V, Balamuralidhar P (2015) Secure MQTT for Internet of Things (IoT). In: 2015 fifth international conference on communication systems and network technologies (CSNT). IEEE, pp 746–751
84. Soroush H, Salajegheh M, Dimitriou T (2007) Providing transparent security services to sensor networks. In: Proceedings of the international conference on communications, pp 3431–3436
85. Staudemeyer RC, Pöhls HC, Watson BW (2017) Security & privacy for the Internet-of-Things communication in the SmartCity. In: Designing, developing, and facilitating smart cities: urban design to IoT solutions, chap 7. Springer, pp 109–137
86. Staudemeyer RC, Pöhls HC, Wójcik M (2018) The road to privacy in IoT: beyond encryption and signatures, towards unobservable communication. In: Proceedings of The 7th workshop on IoT-SoS: Internet of Things smart objects and services (WOWMOM SOS-IOT 2018). IEEE Computer Society
87. Staudemeyer RC, Umuhoza D, Omlin CW (2005) Attacker models, traffic analysis and privacy threats in IP networks. In: Proceedings of the 12th international conference on telecommunications (ICT'05)
88. Steinfeld R, Bull L, Zheng Y (2002) Content extraction signatures. In: Proceedings of international conference on information security and cryptology (ICISC 2001), vol 2288. Springer, pp 163–205. https://doi.org/10.1007/3-540-45861-1_22
89. The European Parliament and the Council of the European Union: Directive 1995/46/EC of the european parliament and of the council—on the protection of individuals with regard to the processing of personal data on the free movement of such data (1995)
90. Thubert P (2011) RFC6282—compression format for IPv6 datagrams over IEEE 802.15.4-based networks
91. Tragos EZ, Angelakis V, Fragkiadakis A, Gundlegård D, Nechifor CS, Oikonomou G, Pöhls HC, Gavras A (2014) Enabling reliable and secure IoT-based smart city applications. In: Proceedings of the international conference on pervasive computing and communication workshops (PERCOM'14). IEEE, pp 111–116
92. Vella M (2014) Nest CEO Tony Fadell on the future of the smart home. TIMES Magazine
93. Waidner M (1989) Unconditional sender and recipient untraceability in spite of active attacks. In: Proceedings of advances in cryptology (EUROCRYPT'89). Springer, pp 302–319
94. Waidner M, Pfitzmann B (1990) The dining cryptographers in the disco: unconditional sender and recipient untraceability with computationally secure serviceability. In: Proceedings of the workshop on the theory and application of cryptographic techniques on advances in cryptology (EUROCRYPT '89), vol 89, pp 690
95. Weiser M (1993) Some computer science issues in ubiquitous computing. Commun ACM 36(7):75–84
96. Wolinsky DI, Corrigan-Gibbs H, Ford B, Johnson A (2012) Dissent in numbers: making strong anonymity scale. In: Proceedings of the 10th USENIX conference on operating systems design and implementation, OSDI'12. USENIX Association, pp 179–192
97. Yao X, Chen Z, Tian Y (2015) A lightweight attribute-based encryption scheme for the Internet of Things. Future Gener Comput Syst 49:104–112
98. Zolertia (2015) RE-Mote datasheet. Retrieved from https://github.com/Zolertia/Resources/wiki/RE-Mote