Data Protection for the Internet of Things

Dissertation

zur Erlangung des Grades

Doktor der Wirtschaftswissenschaft

eingereicht an der Fakultät für Wirtschaftswissenschaften
der Universität Regensburg

vorgelegt von:

Santiago Reinhard Suppan

Regensburg

August 2017

Gutachter

Prof. Dr. Günther Pernul

Prof. Dr. Doğan Kesdoğan

This page is left blank intentionally.

Acknowledgments

This thesis and its achievements would not have been possible without the help of many people which have supported and believed in me. During the course of this thesis, I had the pleasure to work (and befriend) with amazing people, while enjoying the incredible support my family provided. I hereby acknowledge these colleagues, friends and family in this (non-exhaustive) list.

Firstly, I want to thank Prof. Günther Pernul for trusting me with this endeavour, for his open and constructive feedback and for guiding me as my supervisor. I also want to thank Prof. Doğan Kesdoğan for the chance to be under his supervision and his feedback.

I am personally very grateful to my industrial supervisor Dr. Jorge Cuéllar who had great trust in me, provided me with guidance, ideas, knowledge, constructive critique and with whom I had so many interesting discussions. I feel very lucky to have been able to accomplish this work under his guidance and to be able to call him a friend.

I want to thank my family and friends for the incredible support and patience they have provided me with. I am particularly thankful to my brother, Dr. Gottfried Suppan, who has always been my role model, my parents, Carmen and Gottfried Sr., who have always endorsed me in my decisions and my parter, Christina Kelm, who has accompanied me the best part of my life, has motivated and supported me to finish this thesis day by day, weekend by weekend and mid-night by mid-night.

I also acknowledge that the research leading to some of the results presented in this thesis was only possible through collaboration with the European Unions Seventh Programme for research, technological development and demonstration under grant agreement n^o609094. I want to thank all the Rerum partners for their discussions, feedback and motivation (mainly in form of weekend deadlines). It was a wonderful experience to work with you.

I want to thank the Siemens AG for offering me the opportunity and the resources to reach this achievements. I also want to thank all the Siemens colleagues whom I had the pleasure to work with.

Abstract

The Internet of Things (abbreviated: “IoT”) is acknowledged as one of the most important disruptive technologies with more than 16 billion devices forecasted to interact autonomously by 2020. The idea is simple, devices will help to measure the status of physical objects. The devices, containing sensors and actuators, are so small that they can be integrated or attached to any object in order to measure that object and possibly change its status accordingly. A process or work flow is then able to interact with those devices and to control the objects physically. The result is the collection of massive data in an ubiquitous form. This data can be analysed to gain new insights, a benefit propagated by the “Big Data” and “Smart Data” paradigms.

While governments, cities and industries are heavily involved in the Internet of Things, society’s privacy awareness and the concerns over data protection in IoT increase steadily. The scale of the collection, processing and dissemination of possibly private information in the Internet of Things has long begun to raise privacy concerns. The problem is a fundamental one, it is the massive data collection that benefits the investment on IoT, while it contradicts the interest on data minimization coming from privacy advocates. And the challenges go even further, while privacy is an actively researched topic with a mature variety of privacy preserving mechanisms, legal studies and surveillance studies in specific contexts, investigations of how to apply this concepts in the constrained environment of IoT have merely begun.

Thus the objective of this thesis is threefold and tackles several topics, looking at them in a differentiated way and later bringing them together for one of the first, (more) complete pictures of privacy in IoT.

The first starting point is the throughout study of stakeholders, impact areas and proposals on an architectural reference model for IoT. At the time of this writing, IoT was adversed heavily by several companies, products and even governments, creating a blurred picture of what IoT really is. This thesis surveys stakeholders, scenarios, architecture paradigms and definitions to find a working definition for IoT which adequately describes the intersection between all of the aforementioned topics. In a further step, the definition is applied exemplary on two scenarios to identify the common building blocks of those scenarios and of IoT in general. The building blocks are then verified against a similar approach by the IoT-A and Rerum projects and unified to an IoT domain model. This approach purposefully uses notions and paradigms provided in related scientific work and European projects in order to benefit from existing efforts and to achieve a common understanding.

In this thesis, the observation of so called cyber-physical properties of IoT leads to the conclusion that IoT proposals miss a core concept of physical interaction in the "real world". Accordingly, this thesis takes a detour to jurisdiction and identifies ownership and possession as a main concept of "human-to-object" relationships. The analysis of IoT building blocks ends with an enhanced IoT domain model.

The next step breaks down "privacy by design". Notably hereby is that privacy by design has been well integrated in to the new European General Data Protection Regulation (GDPR). This regulation heavily affects IoT and thus serves as the main source of privacy requirements. Gürses et al.'s privacy paradigm (privacy as confidentiality, privacy as control and privacy as practice) is used for the breakdown, preceded by a survey of relevant privacy proposals, where relevancy was measured upon previously identified IoT impact areas and stakeholders. Independently from IoT, this thesis shows that privacy engineering is a task that still needs to be well understood. A privacy development lifecycle was therefore sketched as a first step in this direction.

Existing privacy technologies are part of the survey. Current research is summed up to show that while many schemes exist, few are adequate for actual application in IoT due to their high energy or computational consumption and high implementation costs (most notably caused by the implementation of special arithmetics). In an effort to give a first direction on possible new privacy enhancing technologies for IoT, new technical schemes are presented, formally verified and evaluated. The proposals comprise schemes, among others, on relaxed integrity protection, privacy friendly authentication and authorization as well as geo-location privacy. The schemes are presented to industry partners with positive results. This technologies have thus been published in academia and as intellectual property items.

This thesis concludes by bringing privacy and IoT together. The final result is a privacy enhanced IoT domain model accompanied by a set of assumptions regarding stakeholders, economic impacts, economical and technical constraints as well as formally verified and evaluated proof of concept technologies for privacy in IoT.

There is justifiable interest in IoT as it helps to tackle many future challenges found in several impact areas. At the same time, IoT impacts the stakeholders that participate in those areas, creating the need for unification of IoT and privacy. This thesis shows that technical and economical constraints do not impede such a process, although the process has merely begun.

Contents

List of Figures **v**

List of Tables **viii**

List of Algorithms **viii**

List of Acronyms **x**

1 Introduction **1**
1.1 Motivation 5
1.2 Approach 7
1.3 Outline 10

2 Framing the Internet of Things **12**
2.1 Commercial Relevance and Impact Areas 13
2.2 Differences in Data Generation between Web and IoT Systems . . 17
2.3 Methodology of Identification of Building Blocks for IoT 18
2.3.1 Definition 20
2.3.2 Building Blocks 22
2.4 Architecture Reference and Domain Model for IoT 27
2.4.1 Related Work 27
2.4.2 IoT-A Architectural Reference Model 29
2.4.3 Domain Model 31
2.4.3.1 Rerum Domain Model 35
2.5 Further Architectural Concepts 38
2.5.1 Federation, Collaboration, Cooperation and Composition . 39
2.5.2 Communication Schemes in IoT 43
2.5.3 Ownership and Social IoT 47
2.5.3.1 Ownership Representation in IoT 48
2.5.3.2 Related Work 49
2.5.3.3 A User-centric Implementation for Ownership . . 50
2.5.3.4 Proposal on the Establishment of Ownership . . . 51

2.5.3.5 Ownership in the Rerum Domain Model 57
2.5.4 Implementation of the Rerum Domain Model 59
2.6 Technologies for IoT 61
2.6.1 Economical Value and Size Constraints 62
2.6.2 Classification of IoT Devices and Networks 64
2.6.3 IoT Protocol Layer Stack - IETF LLN 65
2.6.4 IoT Protocol Layer Stack - IETF LLN + HIP 66

3 Privacy in the Internet of Things 70
3.1 Stakeholders 72
3.2 Privacy Engineering 74
3.2.1 Privacy Principles 74
3.2.1.1 The European Data Protection Rules 75
3.2.1.2 European Data Protection Rules - Review 2012 . 77
3.2.1.3 Privacy by Design 82
3.2.1.4 PRIPARE 83
3.3 A Privacy Development Lifecycle 85
3.3.1 Education of System Developers 86
3.3.2 Phase 1 - Purpose Definition and Data Minimization . . . 86
3.3.3 Phase 2 - Threats and Risks Evaluation 87
3.3.4 Phase 3 - Design 89
3.3.5 Phase 4 - Implementation 91
3.3.6 Phase 5 - Verification 92
3.3.7 Phase 6 - Release of System and Education of Stakeholders 92
3.3.8 Phase 7 - Response 93
3.3.9 Challenges of Privacy Engineering in IoT 94
3.3.9.1 Best Practices for Privacy Engineering 94
3.3.9.2 Cognitive and Structural Problems 95

4 Privacy Enhancing Technologies for the Internet of Things 101
4.1 Categorization of Privacy Enhancing Technologies 102
4.2 Privacy Enhancing Technologies Supporting Practice and Control 103
4.2.1 Privacy Dashboard 104
4.2.1.1 Related Work 105
4.2.1.2 Rerum Dashboard 107
4.2.1.3 Integration in the Rerum Domain Model 108
4.2.2 Consent Manager 113
4.2.2.1 Related Work 113
4.2.2.2 Integration in the Rerum Domain Model 116

4.2.3 Privacy Friendly Access Control 119
4.2.3.1 Related Work 121
4.2.3.2 Other Authentication Protocols 130
4.2.4 Sticky Policies for Data in Transit 132
4.2.4.1 Integration in the Rerum Domain Model 133
4.2.5 Privacy Enhancing Tokens 136
4.2.5.1 Efficient Privacy Friendly Authorization 136
4.3 Privacy Enhancing Technologies supporting Confidentiality 152
4.3.1 Position Hiding in Floating Traffic Observation 152
4.3.1.1 Floating Car Observation 152
4.3.1.2 Related Work 154
4.3.1.3 Description of the Scheme 156
4.3.1.4 Generation of Vectors 156
4.3.1.5 Stop at Geodic/Civic Area defined by Policy . . . 158
4.3.1.6 Automatic Stop at Side-Roads by Default 160
4.3.1.7 Example of User Opt-Out with two Active Vectors 160
4.3.1.8 Avoiding Correlation between Vectors 161
4.3.1.9 Privacy Considerations 161
4.3.1.10 Summary and Future Work 162
4.3.2 Pseudonyms 163
4.3.2.1 Related work 163
4.3.2.2 Existing Fundamentals 166
4.3.2.3 Virtually Unlimited Generation of Values 167
4.3.2.4 Choosing Adequate Pseudonyms 169
4.3.2.5 Definition of Path and Jump 170
4.3.2.6 Optimization 170
4.3.2.7 Changing Pseudonyms 171
4.3.2.8 Pseudonym Agreement 172
4.3.2.9 Summary and Future Work 174
4.3.3 Malleable Message Authentication Codes 175
4.3.3.1 Introduction to Homomorphism 176
4.3.3.2 Related Work 178
4.3.3.3 Setup 180
4.3.3.4 MallMAC Scheme - Fundamentals 182
4.3.3.5 The MallMAC Scheme 185
4.3.3.6 Variations of the Scheme 190
4.3.3.7 Protocol Verification 193
4.3.3.8 Summary and Future Work 194
4.3.4 Group Message Authentication Codes 194

4.3.4.1 Introduction to Group Signatures 195
4.3.4.2 Related Work 198
4.3.4.3 Setup 200
4.3.4.4 The GroupMACs Scheme 200
4.3.4.5 Considerations and Optimization of Group MACs 207
4.3.4.6 Protocol Verification 208
4.3.4.7 Summary 209

5 A Privacy Enhanced IoT Domain Model 210
5.1 Rerum Pre-Final Domain Model 210
5.1.1 Trust and Privacy in IoT 212
5.2 The Final RERUM Domain Model 214
5.3 A Privacy Enhanced IoT Domain Model 216
5.4 Summary 218

6 Conclusion 219
6.1 Outlook 221

Appendices 224

A Assumptions on the Virtual Representation of Users 225

B Protocol Design and Verification 227
B.1 Design of Efficient Protocols 227
B.2 On the Verification of Security Protocols 228
B.2.1 The Scyther Formal Model 231
B.3 Verification of the MallMACs Protocol 233
B.4 Verification of the GroupMACs Protocol 238

Bibliography 242

List of Figures

1.1 Methodology overview 8

2.1 Generation of data in [a] web based systems and [b] IoT 18
2.2 From a definition and building blocks to a common architectural reference 19
2.3 A Smart Home Scenario (left) and a Fitness Tracker Scenario (right) 23
2.4 The same building blocks could be identified in both scenarios . . 25
2.5 From scenarios (top) to architectures (bottom left) to building blocks and an architectural reference model (bottom right) 26
2.6 The IoT-A: internet of things domain model 32
2.7 Rerum first phase ARM 37
2.8 Contributions for Privacy Enhancing Technologies in Rerum . . . 38
2.9 Cloud based communication scheme 44
2.10 Gateway based communication scheme 45
2.11 Intranet based communication scheme 45
2.12 Distributed communication scheme 46
2.13 Ownership scenario - setup 52
2.14 Ownership scenario with enabling technologies 54
2.15 Ownership and digital shadow 55
2.16 Ownership protocol example 56
2.17 Integration of ownership in the Rerum domain model 58
2.18 Rerum middleware 61
2.19 Comparison of the TCP/IP stack (left) and the IETF IoT LLN stack (right) 65
2.20 IP based communication 67
2.21 HIP based (bottom) 68

4.1 Draft for the Rerum dashboard [SWC+15] 108
4.2 Interaction of Privacy Dashboard and Consent Manager 109
4.3 Interaction of Privacy Dashboard and Anonymising and Pseudonymising Manager 110

4.4 Interaction of Privacy Dashboard and Activator / Deactivator of Data Collection . . . 112
4.5 Sequence of the acquisition of user consent . . . 117
4.6 Rerum Consent Manager: Example for consent request in the UC-I2 trial . . . 119
4.7 The access control modules (outlined) of the Rerum architecture . 120
4.8 A simple Sticky Policy mechanism . . . 122
4.9 Exemplification of XACML by mapping a real life example . . . 123
4.10 Integration of XACML components in the Rerum middleware . . 125
4.11 Sequence for the access of protected data in Rerum through XACML 126
4.12 Sequence of the OAuth 2.0 protocol . . . 128
4.13 Sequence of the DCAF protocol . . . 129
4.14 Sticky Policies in the Rerum domain model . . . 134
4.15 Pseudonym-based authorization tokens . . . 140
4.16 Energy profile comparison of different cryptographic functions [Che15] 149
4.17 Geo-location data in traffic measurement . . . 154
4.18 Time controlled vectors . . . 157
4.19 Stop at geodic/civic area . . . 159
4.20 Stop at side roads . . . 160
4.21 Hash-Tree with an initial input x . . . 168
4.22 Selection of adequate outputs as pseudonyms . . . 169
4.23 Example of a pseudonym agreement for data retrieval . . . 173
4.24 Example of re-constructing a pseudonym by a device owner . . . 174
4.25 Possible secrets and paths for Signer, Sanitizer and Verifier . . . 186
4.26 The MallMAC Protocol . . . 188
4.27 Proposed variation for sanitization knowledge by using different secrets . . . 190
4.28 Variation of the distribution of secrets for better revocation . . . 193
4.29 Variation of the distribution of secrets for better revocation . . . 195
4.30 Example of existing hash trees in the scheme with two Signers . . 202
4.31 Example of a verification table . . . 203
4.32 Simplified sequence diagram for the proposed scheme . . . 204
4.33 Verification of two messages in two different time slices . . . 206

5.1 Rerum domain model - six new extensions . . . 211
5.2 Rerum final domain model . . . 215
5.3 A Privacy Enhanced Domain Model for IoT . . . 217

B.1 Scyther - Basic formal model. . . . 231
B.2 Scyther - Protocol execution . . . 231

B.3 Scyther - Matched typing and bounds 232
B.4 Verification of the MallMACs protocol 237
B.5 Verification of the GroupMACs protocol 241

List of Tables

2.1 Comparison of size in relation to costs 64
2.2 IETF classification of constrained devices 64

4.1 Comparison of energy and computational profiles of AA schemes . 150
4.2 Evaluation of AA schemes for constrained devices in lossy networks 151
4.3 Generation of multiple vectors 157
4.4 Multiple active vectors before entering an opt-out area 161
4.5 Averaged vector sent at entrance of an opt-out area 161
4.6 Group MACs: Initial distribution of secrets 201

List of Algorithms

1 Construction of functions G, g1, g2, g3, g4. 143
2 Construction of main data structure used by the function G . . . 145
3 Example generation of a value for G 145
4 Traversing the PAT data structure 146
5 Initial creation of St 147
6 Creation of Ct 147
7 Generation of HMAC values 167
8 Creating pseudonym data structure 171
9 Generation of keys in a Group Signature Scheme. 196
10 Distribution of keys in a Group Signature Scheme 196
11 Adding new members in a Group Signature Scheme 196
12 Revoking members in a Group Signature Scheme 197
13 Sign a message in a Group Signature Scheme 197
14 Verify a message in a Group Signature Scheme 197
15 Re-link (or “opening”) a Signature in a Group Signature Scheme . 198
16 Actions of Signer 1 204
17 Actions of Verifier 1 205

List of Acronyms

IoT Internet of Things . . . 1

ICT Information and Communications Technology . . . 10

IERC European Research Cluster on the Internet of Things . . . 21

IoT-A Internet of Things Archtiecture Project . . . 26

PET Privacy Enhancing Technology . . . 10

PETs Privacy Enhancing Technologies . . . 6

OSN Online Social Network . . . 5

PbD Privacy by Design . . . 82

PDLC Privacy Development Lifecycle . . . 11

DFDs Data Flow Diagrams . . . 87

FAQ Frequently Asked Questions . . . 92

EMS Energy Management System . . . 24

CoAP Constrained Application Protocol . . . 28

6lowPAN IPv6 over Low power Wireless Personal Area Network . . . 28

OSI model Open Systems Interconnection model . . . 28

ARM Architectural Reference Model . . . 27

WSN Wireless Sensor Networks . . . 28

PE Physical Entity . . . 33

VE Virtual Entity . . . 33

AE Augmented Entity . . . 33

VRD Virtual Rerum Device . . . 35

GVO Generic Virtual Object . . . 60

WWW World Wide Web . . . 17

IETF Internet Engineering Task Force . . . 9

LLN Low Power and Lossy Networks . . . 65

ROLL Routing Over Low power and Lossy networks 65
HIP Host-Identity Protocol 53
ID Identifier 33
IESG Internet Engineering Steering Group 69
IEEE Institute of Electrical and Electronics Engineers 65
DTLS Datagram Transport Layer Security 68
NFC Near Field Communication 21
RFID Radio-Frequency Identification 21
GDPR General Data Protection Regulation 77
OECD Organisation for Economic Cooperation and Development 82
UCC Uniform Commercial Code 99
SDLC Security Development Lifecycle 94
GVO General Virtual Object 60
FOI Feature Of Interest 60
SPT Rerum Security, Privacy and Trust Components 107
HC Health Care 114
XACML Extensible Access Control Markup Language 99
PEP Policy Enforcement Point 120
PDP Policy Decision Point 124
RMW Rerum Middleware 59
PAP Policy Administration Point 120
OAuth Open Authorization standard 121
SAM Server Authentication and Authorization Manager 137
CAM Client Authentication Manager 138
RO Resource Owner 127
AS Authentication Server 127
AG Access Grant 128
C Client 127
S Server 138
RS Resource Server 127
ST Server Token 138

CT Client Token . . . 138

AT Authorization Token . . . 139

DCAF Delegated CoAP Authentication and Authorization Framework 128

TTP Trusted Third Party . . . 122

AM Authentication Manager . . . 129

DTLS Datagram Transport Layer Security . . . 68

SAML Security Assertion Markup Language . . . 121

PAT Pseudonym-based Authorization Tokens . . . 138

PRG Pseudo-Random Generator . . . 143

ACE Authentication and Authorization for Constrained Environments IETF Working Group . . . 136

AA Authorization and Authentication . . . 148

ECQV Elliptic Curve Qu-Vanstone Implicit Certificates . . . 148

TA Trusted Authorities . . . 135

3DES Triple-Data Encryption Standard . . . 148

AES Advance Encryption Standard . . . 148

ECDSA Elliptic Curve Digital Signature Algroithm . . . 148

CA Certificate Authority . . . 131

VANETs Vehicular Ad-Hoc Networks . . . 153

BASt German Federal Highway Research Institute . . . 153

GGM Goldreich, Goldwasser and Micali . . . 164

MAC Message Authentication Code . . . 166

HMAC Keyed-Hash Message Authentication Code . . . 166

SSS Sanitizable Signature Scheme . . . 175

MallMACs Malleable Message Authentication Codes . . . 177

RSU Road-Side Unit . . . 198

GroupMACs Group Message Authentication Codes . . . 200

Necessity is blind until it becomes conscious. — Freedom is the consciousness of necessity.

Karl Marx

Chapter 1

Introduction

The Internet of Things (IoT) is expected to bring massive changes upon society and economy over the next 10 to 15 years [SGFW10]. The IERC [VFG+11] describes IoT as: "*a dynamic global network infrastructure [...] where physical and virtual "things" have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network.*"

This global network of "things" is growing steadily with more than 16 billion devices forecasted to interact by 2020, see [VF10]. Devices will be able to gather data from different objects in different contexts in a higher quality and quantity than ever before. The data will then be processed by service providers worldwide through cloud services and mobile programs (or "apps") which are estimated to reach more than 253 billion in 2017, see [Sta15]. In order to understand why data is an asset of such a great value, it has to be understood how economical value can be generated from it. Kenneth Cukier, editor of "The Economist" and previous editor of "Wall Street Journal Asia" expresses it as follows (see [Cuk14]): *"Well, think about it. You have more information. You can do things that you couldn't do before. [...] The general idea is that instead of instructing a computer what do, we are going to simply throw data at the problem and tell the computer to figure it out for itself. [...] And this idea of machine learning is going everywhere."*

The idea described by Kenneth Cukier is reflected in the Big and Smart Data boom. Data is stored in the cloud, linked to other data (form other sensors or other systems) and the "enriched" data can reveal structures that give new insights

and allow new business opportunities. This new structures correspond to purposes not identified in the first place which *fundamentally* contradicts European (and U.S. American) data protection regulations, which have at their core user consent and data processing that is bound to a purpose, see [Sol12].

A common understanding within IT companies is that parties collecting the data (which are normally not the person to whom the data refers to), are free to decide what they want to do with it. This point of view has been brought to public focus recently by automobile companies which discussed whom the data in connected cars belong to. The argument was that companies which produced single car components (e.g., breaks, relays, bus systems) created and claimed the data, while car manufacturers claimed the data because those components were used in the car itself[1]. The discussion left the driver out completely, whose behaviour is measured by the car sensors. This contradicts another fundamental privacy principle: it is neither the "data generator" nor the "data controller", the entities that should determine how the data can be used and by whom. The entity responsible for those decisions is solely the driver or "data subject" [2].

These misconceptions and challenges for data protection in IoT have long begun to raise privacy concerns in the research community (for instance, see [May09a] [Web10a] [AIM10a] [MS10]). Also, the European Commission has expressed concern regarding the challenges of data protection in the Internet of Things, see [TEGIE13], as well as Data Protection and Privacy Commissioners, see [KM14].

But Data Protection and Privacy Commissioners also underline in [KM14] that IoT and data analysis *are useful* for society. For example, health care has benefited significantly from exchanging sensitive health records from patients, see [WPJ+05]. With the data from patients in different contexts, further health care support can be developed, for example in the case of assisted living. "Public data mining" in cities can support efficient waste management, reduction of costs

[1] For a snapshot of the discussion, see [DeB15].

[2] The terms data subject, data controller and data generator are specific terms to describe whom the data belongs to, who generated it and who processes, stores or maintains it. They are commonly used in privacy, e.g. in the European General Data Protection Regulation (GDPR). For the sake of simplicity, this thesis will use the terms *user* as synonym for data subject and *service provider* as a general collective term for data generator and controller.

related to traffic and logistics as well as a raise in comfort quality regarding air and noise pollution.

The Article 29 Data Protection Working Party states that society's acceptance and the benefits promised to cities and industries regarding the potential of the Internet of Things will heavily depend on the perception of data protection provided by IoT. The following quotation is taken from [EU 14]:

"*Organisations which place privacy and data protection at the forefront of product development will be well placed to ensure that their goods and services respect the principles of privacy by design and are equipped with the privacy friendly defaults expected by EU citizens.*"

Privacy in the IoT context is not only a matter of "avoiding data" as data collection and analysis is beneficial. The problem with this idea is that companies may not know how to benefit from data, thus collecting large amounts until they understand how to use it. As the purpose is unknown by the time of collection, data subjects may not be informed properly or they are drawn into consenting to processing policies that they do not understand for the immediate benefit of a service or even for the lack of alternatives[3].

Hoarding data may be risky, as shown in the Cost of Data Breach Study, see [LLC15]. The study shows that a single lost or stolen record may cause an average damage of $154. Data breaches ranged in 2015 from 22000 to more than 100000 lost records per incident, equalling an economic damage from $338000 to $15 million. This does not include "mega breaches" which, according to the study, disclose a minimum of 10 million records, causing a damage of 1.3$ billion or more. The costs per breach include direct and indirect costs. Direct costs incur engaging forensic experts, hotline support and providing discounts to recompense the user. Indirect costs are extrapolated calculations of customer loss and diminished customer acquisition.

But in some cases the consequences for companies may not be as big as for the users and therefore the motivation to avoid unnecessary collection or to adequately protect private data may be low. For example, the data breach of Sony's Playstation Network in 2014 was estimated to cost more than $100 million,

[3] As the introductory quote says, *"Necessity is blind."*

see [Cor14]. In their following quarter financial report, Sony stated that they could reduce the costs to $15 million, see [Son15]. Additionally, Sony was able to use the attention from the breach to push their publicity, flagging the attack as an attempt to stop the release of a controversial movie produced by Sony, thus gaining more subscribers as Sony released the movie in their entertainment network exclusively, see [Dea15]. In summary, Sony could shift the attention from the privacy breach to attract users to their network and compensate their losses (or even gain benefit from it due to savings in advertisement), while the lost data records, providing information about millions of user credit cards, passwords and payment details of Sony employees remain in the hands of the attackers, see [Mus11].

A serious consequence that may come along data breaches or surveillance of individuals is the so called *chilling effect*, see [Sol07]. The chilling effect describes the fear of individuals of being judged by their actions and decisions. In order to avoid such effects and increase user acceptance, IoT system operators will have to be transparent to users, for example, by showing that their systems are not used for extensive tracking. In cases of a data breach where information that can be used to judge individuals is lost, the consequences are unpredictable.

In order to push companies to protect user data, high sanctions as well as regulations have been demanded that do not allow loopholes. These exist in current directives, as they are too high-level to decide whether protection is adequate or not, see e.g. [Web09]. Accordingly, there is need for technology to protect personal data in different scenarios. For example, if no efficient Privacy Enhancing Technology (PET) is available for IoT devices due to their constraints, only a best effort protection can be demanded from companies, even if it is inadequate. There are several other challenges faced by all the stakeholders of IoT sucha s Internet of Things developers, data protection authorities and individuals, as stated in [KM14]. For example, the data gathered in public spaces relates to many different individual data subjects, and it will not be easy to ask them for consent or inform them about the purpose of collection. The transparency of the data provenance and integrity is difficult to guarantee in a scenario where subjects are continuously being monitored and tracked by a large number of devices. And even if technical solutions exist, constrained devices used in IoT make the use of

security and privacy mechanisms difficult to implement, configure and use. This challenges and their interdependencies will be analysed during the course of this thesis.

1.1 Motivation

This thesis provides an understanding of the fundamental conflicts of privacy and IoT, which were briefly introduced in the Chapter 1. This is done by reconciling existing work and serving new insights as a starting point to privacy in IoT by showing which problems and challenges are the most imminent.

Although this thesis follows many technical proposals and explores how these technical methods can be applied upholding the constraints imposed by IoT devices, it does not focus on the specification of the "right Bits" for single technologies. Developed protocols are means to understand why a specific technology makes sense in the IoT context.

The first main challenge that this thesis wants to address is to identify the *building blocks of the Internet of Things*. The existence of many road-maps, definitions, whitepapers and architectures exist which make it difficult to define what IoT actually is and how it is different from known systems like web systems or sensor networks. It has to be understood upon which blocks the IoT will be built on and which technologies will be used to bring an IoT system to life, for example how proposals like [GBMP13], [RNL11] and [BBF+13] help to create a common IoT architecture and how technologies like [Har12], [SHBF12], [Mul07] and [GNC+01] may enable IoT but potentially limit data protection.

Protecting the privacy of system participants as well as casual users and non-involved subjects in a future IoT is the second main challenge in privacy research. The problem is the ubiquitousness and pervasiveness of sensors that measure the status and context of environments. This will affect individuals regardless of them being a user of the system or not. Data collection in IoT differs therefore from other systems that are considered to be potentially intrusive, such as Online Social Networks (OSNs). Online Social Network (OSN) generally trade privacy for commodity and require active user interaction to consume the user's data. Data

protection in IoT has therefore many aspects that need to be analysed and other research fields may need to be incorporated as well for possible answers, examplse being jurisprudence[4] and health care. Additionally, the constrained environment of IoT does not allow to use existing Privacy Enhancing Technologies (PETs), thus new PETs or adaptations have to be developed.

The last main challenge is enabling privacy preserving services in IoT. This may be possible, if data protection can be deeply routed into an IoT system such that privacy and value of data can be balanced. But there is no consensus on how privacy can be engineered into a system, let alone into a network of systems such as IoT. Many taxonomies, principles and approaches exist for building privacy into a system.

Derived from the identified challenges above, this thesis will analyse following questions:

Identifying the building blocks of IoT

(a) *Which are the existing efforts and proposals to support a future IoT?*

(b) *Which are the main building blocks that IoT needs?*

(c) *Which building blocks are specially relevant for privacy in IoT and which building blocks have to be further defined to allow "privacy by design"?*

Data protection in IoT

(a) *What requirements do European Data Protection Regulations impose on practical IoT, particularly the new proposals of [C+ 12b]?*

(b) *Which existing privacy taxonomy can be applied to IoT?*

(c) *Which existing privacy enhancing technologies are relevant for IoT?*

(d) *How can existing privacy enhancing technologies be adapted or applied in concrete IoT scenarios?*

Privacy by Design in IoT

(a) *What does "Privacy by Design" mean in the context of IoT?*

(b) *How can identified privacy enhancing technologies enable privacy by design?*

[4] Jurisprudence is the "the study, knowledge, or science of law", see [Cor16].

(c) *How does the resulting privacy friendly IoT architecture reflect privacy by design?*

1.2 Approach

This thesis follows the design science research methodology from Peffers et al. [PTRC07], based on the original proposal by Hevner et al. [HMPR04]. The proposed methodology by Peffers et al. comprises of six activities:

Problem identification and motivation. Problem identification was introduced in Section 1.1 and is further elaborated on related work discussed per Chapter.

Definition of the objectives for a solution. Objectives or research questions are defined in Section 1.1. Additional objectives are derived from the Rerum project, which this thesis accompanies. The objectives will be referenced through the thesis for clarity.

Design and development. Peffers et al. refer to the creation of artifacts such as "constructs, models, methods, instantiations or new properties of technical, social, and/or informational resources" to answer research questions. As data protection in IoT covers a wide variety of problems, this thesis will provide many different artifacts, such as an extended architectural reference model in Section 2.3.1, protocols to show the viability of PETs in constrained environments in Section 4.2 as well as a privacy development lifecycle in Section 3.3 as a proposed consensus on privacy engineering.

Demonstration. The variety of aritfacts will be demonstrated in use cases that are based on the Rerum project. Hevner et al. underline that artifacts should target "unsolved and important business problem[s]", therefore artifacts of this thesis were published in the industry context as collaborative patents and other intellectual property instruments with Siemens AG. References to those publications are cited in the respective Chapter.

Evaluation. Artifacts were evaluated through formal verification, publications and also in laboratory experiments. The evaluation form will be described per artifact in its respective Chapter.

Communication. Artifacts in this thesis were communicated to project and industry partners as well as standardization bodies. Details will be described per Chapter.

Figure 1.1 displays the relationship between the research method, related work and results of this thesis. The individual relations between related work, methodology and resulting contributions are explained in the respective Chapters.

For problem identification and motivation, this thesis will use, adapt and extend the use cases of Rerum, see [RER14a].

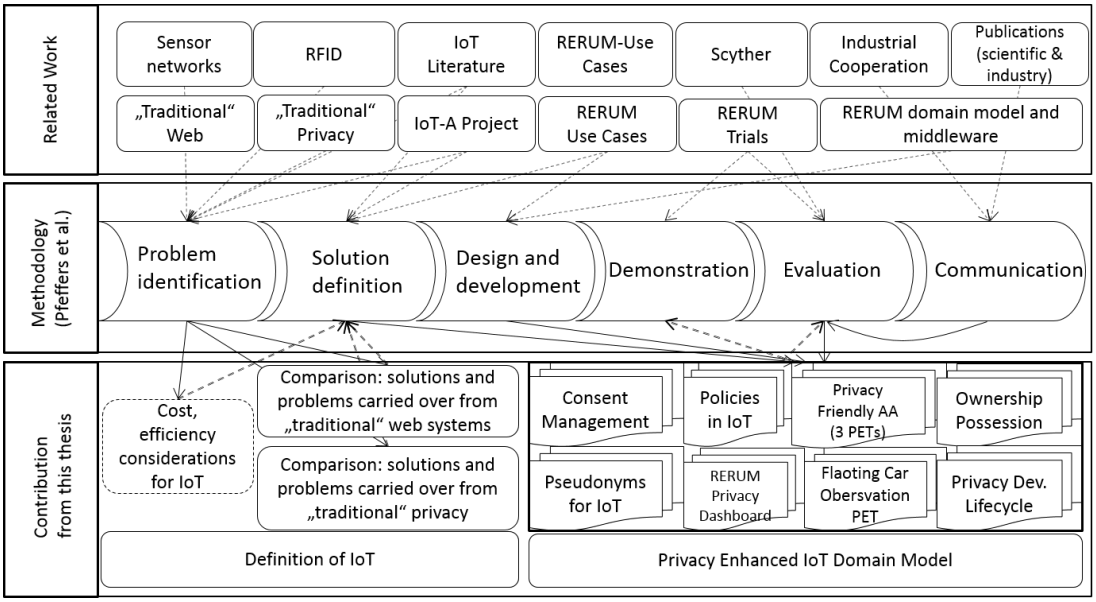

Figure 1.1: Methodology overview

The main contributions of this thesis are:

- *Survey of traditional web systems and their evolution to the Internet of Things.* As a clear cut definition is inexpedient, this survey focuses on structures and problems from the traditional internet that are carried over to the Internet of Things. The survey is firstly published in this thesis.
- *Survey of privacy in traditional web systems, cognitive and structural problems and how they carry over to the Internet of Things.* Solutions are presented in form of related work and own proposals like a privacy development life-cycle, architectural concepts and knowledge transfer from other scientific fields, namely jurisprudence and medicine, and the technical artifacts listed below. The survey is firstly published in this thesis.

- *Consent management* is one of the structural problems of traditional data protection. This artifact describes a technical and conceptual solution to the scaling problem of consent. The problem derivation and definition is only found in this thesis. The architectural artifact was developed for this thesis, presented and published within the Rerum project.
- *Policies in IoT* are a fundamental technology to support consent management. This technical artifact was developed for this thesis and was published as an intellectual property item[5] by Siemens AG in 2014, see [SC15]. The artifact was thereafter introduced to and published by the Rerum project.
- *Privacy friendly authentication and authorization* are other fundamental technologies needed for data protection. The technologies analysed were based on the requirements of the Rerum project's use cases. The resulting artifacts comprise PETs for Malleable Authentication Codes, Group Message Authentication Codes and Privacy Friendly Authentication Tokens. The Malleable MACs and Group MACs have been adopted by Siemens AG as intellectual property items[6]. Privacy enhanced tokens have been published as an Internet Engineering Task Force (IETF) draft and are further developed by Siemens AG independently, see [CSP15].
- *Ownership and Possession* are fundamental elements of German law. Ownership and possession are useful for data protection to be able to differentiate the privacy policies that apply in case a physical object that senses data is used by a new subject. This artifact adopts the definition of German law and transfers it to data protection. This artifact has, to the best knowledge of the author, no related work and is firstly introduced in this thesis.
- *Pseudonym management* is used to generate and agree (for a limited time) to use pseudonyms between different parties in IoT. This technical artifact was developed for this thesis and was published as an intellectual property[7] item by Siemens AG in 2014.
- *Rerum Privacy Components.* This artifact describes the components of the

[5]Under the official patent file reference number 102015202769.4.

[6]Under the official patent file reference numbers DE 102015205111 A1 and 102015211932.7, respectively. Authors are Santiago Suppan and Jorge Cuéllar, filing by Siemens AG.

[7]Under official patent file reference number DE 102015203543 A1. Authors are Santiago Suppan and Jorge Cuéllar, filing by Siemens AG.

Rerum architecture that have been developed in the course of this thesis and in cooperation with the Rerum project. The components comprise methods for intervenability (temporary revocation of consent), transparency (privacy dashboard) and use case specific archtiecture (floating car observation for Rerum use case UC-O1.)

- *Geo-location Privacy Technology for Floating Car Observation.* This artifact explains why related work on location privacy is too heavy and how a light weight geo-location Privacy Enhancing Technology (PET) for floating car observation could look like.
- *A Privacy Development Lifecycle.* This artifact suggests a privacy development lifecycle and shows how it can be combined with a security development lifecycle for synergies. This artifact has been submitted in 2015 and is going to appear 2016 in [SC16].
- *A Privacy Enhanced IoT Domain Model.* This artifact summarizes all previous artifacts in an IoT domain model. The domain model sets all privacy and IoT building blocks into relation and is a key contribution of this thesis.

Every artifact is accompanied by a literature review to further understand the problem, underline motivation and draw conclusions. The literature reviews follow Webster and Watson's approach, as seen in [WW02].

1.3 Outline

The outline of this thesis is as follows:

In Chapter 2 an introduction to IoT is given. The Chapter begins with a literature review to understand the motivation behind IoT, its economical value and the most promising building blocks for future IoT research. The Chapter outlines technical standards and architecture proposals, compares them to traditional Information and Communications Technology (ICT) systems and reconciles the building blocks of IoT systems. The building blocks will be used to develop privacy enhancing technologies in Chapter 4.

In Chapter 3 privacy in the Internet of Things is discussed. Fundamental

conflicts of privacy and IoT are recapitulated, several privacy principles, privacy taxonomies and privacy frameworks are discussed and unified in a proposal for a Privacy Development Lifecycle (PDLC). The Privacy Development Lifecycle (PDLC) is used to identify the needed steps of privacy engineering and to analyse the challenges in engineering privacy in IoT systems.

Privacy Enhancing Technologies are a fundamental part of privacy engineering. PETs are fairly well developed, e.g., PETs that hold integrity conditions in anonymous environments and to allow observability in public networks. In the Internet of Things many of the current PETs are either too inefficient, economically unsustainable or have not been researched thoroughly. Chapter 4 elaborates on the PETs that are needed in IoT, based on scenario descriptions of the Rerum project and the lessons learned of the PDLC proposal. This Chapter also presents a proof of concept of several artifacts, such as model based integration of policies, achieving malleability properties of message authentication codes, a lightweight, privacy preserving authorization protocol based on symmetric cryptography and a message authentication code scheme that resembles the properties of group signatures.

Chapter 5 resumes the presented concepts and technologies, building blocks and domain model proposals in a *privacy enhanced IoT domain model*.

Chapter 6 concludes this thesis and gives an outlook towards open challenges and future work for data protection in IoT.

Chapter 2

Framing the Internet of Things

There is an undeniable hype surrounding the Internet of Things. Companies with a consulting focus like McKinsey, Accenture and Gartner state that IoT is the most important technology trend of the next years, see [Com15b, DBNA14, JKW15]. But also companies with a technical focus like, Google[1], Microsoft[2], Siemens[3] and others[4] show that IoT promises competitive advantage. One problem is, that public relations and marketing strategies have developed many definitions using terms like "web of things", "web of systems" and "web of everything" as synonyms. It is therefore that a lot of confusion exists on what the Internet of Things actually is and how it differentiates itself from traditional IT systems which have been simply called "IoT" or "smart" to follow the trend.

In this Chapter different definitions of IoT will be discussed to adopt a working definition for the rest of this thesis. As IoT is still a fairly recent topic, related efforts of standardization bodies will be emphasized to make sure that the adopted definition is long-lived. The remainder of this Chapter is outlined as follows: Section 2.1 captures the current status of IoT in society and economy. Section 2.2 compares traditional ICT systems and IoT systems and underlines similarities and differences. Section 2.3 elaborates a methodology to find building blocks for IoT, Section 2.3.1 particularly discusses the different visions and definitions of

[1] Google wants to provide "the" operating system for the IoT, see [Wee16].

[2] Microsoft wants to unify IoT devices with their Windows 10 operating system, see [Mic16].

[3] Siemens manufactures a wide range of products and services for several application fields including smart cities, industries and mobility. Additionally, Siemens wants to provide services for their products to cover the whole IoT lifecycle, see for example the explanation in [Weg15].

[4] See e.g. [IBM16, Int16, Ama16a].

IoT. The Section concludes with the definition that is going to be used for the rest of this thesis. Section 2.4 introduces the IoT-A and Rerum architectural reference model and identifies the most important building blocks form the adopted definition. Section 2.4.2 defines the IoT terminology that will be used in the following Chapters, including definitions of a common domain model and an architectural reference. Section 2.5 describes additional architectural concepts and communication types in IoT. This concepts come into play in the design decisions for PETs in Chapter 4. Section 2.6 reviews the proposed technologies to support the identified building blocks and architecture. This Section concludes with a set of "enabling technologies of IoT" which will be used in Chapter 4 to show how these technologies favour and limit PETs.

2.1 Commercial Relevance and Impact Areas

The term "Internet of Things" is said to be coined by Kevin Ashton in 1999, see [Ash09], when he linked the idea of RFID tags with the problem of a huge amount of heterogeneous goods of Proctor and Gamble, in order to analyse and optimise the company's supply chain. The Internet of Things was born as a business solution and it has been adopted as such, as is reflected in the opinions of governments world wide, for example:

The Council of the European Union mentions the importance of IoT in March 2008 in [EU14], where it recognizes the Internet of Things as *"poised to develop and to give rise to important possibilities"* and at the same time it acknowledges that IoT *"represents risks in terms of the protection of individual privacy"*.

The United States National Intelligence Council acknowledged its disruptive importance in 2008 [Cou08], stating that IoT may *"contribute invaluably to economic development and military capability"*.

The Chinese government has first stated their interest in IoT in 2009, see [SGFW10], with reported investments of $1.6 billion in 2014 and an estimated market value of $ 163 billion by 2020[5].

One of the most significant technologies for IoT, as discussed in the following

[5]Figures reported by the GSMA, see [GSM15].

Chapter, is RFID. The RFID market is estimated to amount $13.2 billion worldwide[6]. Other building technologies in IoT, which may differ from application field to application field, may amount similar market values with time, making the economical significance of IoT many times larger.

There are several social areas that IoT will have impact on. Miorandi et al. identify six market sectors in [MSPC12] which are **Smart Homes & Buildings**, **Smart Cities**, **Environmental monitoring**, **Health-care** and **Smart Business**. Gubbi et al. additionally determine in [GBMP13] **Utility** and **Mobile**, which will be integrated in to the Smart Cities application field in the following overview.

Smart Homes and Buildings are envisioned as a tool against pollution and gas emissions from cities. The United Nations report on global warming of 2007 revealed that 70% to 80% of the CO_2 emissions are produced in cities. Smart homes and buildings can help to reduce the CO_2 footprint by decentralizing energy management and allowing on-site production of energy and the optimization of energy consumption. The role of IoT here is to manage sensors that constantly overview energy consumption and process the feedback of decentralized energy production. Additionally, IoT helps to align energy optimization strategies with user policies. Smart homes and buildings will also host many "smart appliances" which will interact dynamically to provide commodity services for users such as automatic temperature regulation, order of comestibles if their amount falls below user defined thresholds, safety checks for door and window locks as well as for critical appliances e.g., stove-tops, ovens, water pipes, etc.

Smart City is a term that unifies several application areas within a city. Cities will be facing major challenges, as they are where up to 70% of the world's population is estimated to concentrate, see [Uni14]. One problem that comes with this fact is the energy provision for the urban population. Traditional electrical grids are built to transmit energy in one direction, usually provided by energy producers. As the grid needs to hold a steady electric frequency, high energy demand or low energy demand causes destabilisations that can lead to black outs. With urban population increasing, the current grid structure will not be able

[6]Figures by idtechex, see [DH15].

to cope with the volatility of demand. To handle the raising urban population, a decentralized approach may be helpful. For this, the role of consumers and producers has to merge, allowing households to produce, consume and sell their own energy. Many energy producers will then be spread throughout the cities forming "micro-grids" that balance their own demand and supply. A micro-grid can give or take energy from other micro-grids in case of over- or under-production. This is the so called "self-monitoring" and "self-healing" properties of the future energy grid, also called "smart grid". IoT will support many of the components of the future smart grid, providing a platform for millions of sensors and actuators in homes, electrical nodes and electrical markets which evaluate demand and supply to establish energy prices. Gubbi et al. [GBMP13] point out that utility companies will profit from a network of sensors in remote energy meters (or"smart meters") and water pumps which can provide a real time profile of energy transmission and water networks in order to ensure high quality supply.

Traffic congestions and high pollution rates are other typical problems of cities with high population. The solution is refereed to as "smart transportation": vehicles in a city contribute information (this is called floating car observation) to a service provider, for example the municipality of a city, in order to gather the traffic volume. The service providers then estimate possible congestions in different routes and suggest the fastest detour to the contributing vehicles. The suggested routes also help to distribute the traffic over several routes to avoid congestions. Smart transportation can also include continuous traffic members such as public transportation vehicles or garbage collectors. This type of traffic participants circulate a city based on service requests. Until now, public transport and garbage collectors needed to pass every stop and collection point in a route. In the future, buses and garbage collectors are going to dynamically chose their route to avoid empty bus stops and collection points. The dynamic routes are then enriched by traffic estimation information of the service providers. IoT will again help to provide the platform for all different types of traffic participants, to connect various sensors for floating car observation and to combine different services for a holistic traffic analysis. Gubbi et al. [GBMP13] also list freight

carriers as beneficiaries of smart transportation with intelligent routes based on traffic estimation and dynamic service provision.

Environmental monitoring and comfort quality analysis are similar in their application. The European Union has defined exposure limits for several pollutants such as O_3 (ozone) and CO_2[7] to reduce health risks. IoT can be used to identify other emissions and pollution factors that could have detrimental effects on citizens' health by setting up sensor clusters all over a city which measure different type of factors like temperature, windfall, rain, humidity, river height and others. This application field is called environmental monitoring, which can also be used to react to other scenarios: sensors and actuators may help to recognize an emergency situation and assess the threat level to individuals located in the area.

Exposure limits are generally established to minimize possible health risks. However, environment-related quantities can also be adapted/changed to create individual comfort zones. This kind of services are called comfort quality services. Sensors and actuators are used to analyse the close environment of individuals (temperature, humidity, etc.) and to change that environment (when possible) according to the wishes of the users.

A particular application of comfort quality is **Health Care**. A personal body network and sensor clusters in the proximity of patients can provide a detailed health status to health service providers. The combination of several measurements can help to identify emergency situations, trigger an adequate reaction and even react preventively before the patient reaches an alarming condition. Sensors and actuators can also support elders in their everyday activities (so called assisted living) and health aware citizens to track their activities, such as recognize unhealthy behaviour and optimizing their nutrition.

As stated in the previous Section, the term Internet of Things was born of the idea of using sensors to optimize business processes. **Smart Business** is a collective term for application fields that relate to business and industry, for example inventory management in smart and chaotic warehouses, just-in-time strategies and supply chain management, individual customer care, theft protection

[7]For other different variables, see [Eur08].

and intelligent industries[8] which can be set-up and administrated on short terms, as IoT technology identifies resources, creates relations between them, creates ordering processes for missing resources or retrieves plans to create them on-site (e.g. with 3D printers).

These fields are just a subset of all possible applications of IoT. Although diverse, they all observe users activities either directly or indirectly. In the next Section, a brief overview of related projects and standardization activities is given, which relate to privacy protection in the application fields presented in this Section.

2.2 Differences in Data Generation between Web and IoT Systems

Web based systems and IoT systems have several things in common: they are distributed by nature, their participants share and process a vast amount of data and their technologies constantly adapt to new constraints and environments. IoT has specific characteristics like resource constrained devices and lossy networks, which will be detailed in Section 2.6.2. One of the main differences however, is how data is introduced into each system as shown in Figure 2.1.

Data generation in web based systems ([a] in Figure 2.1) is actively driven by system users. Users either participate in the systems and actively upload and input data or the system generates data according to the users activities. Users interact hereby on clients that have high resources regarding computational power, energy and connectivity. In particular, clients are able to connect to wide area networks such as the World Wide Web (WWW).

[8] Advertised as “Industry 4.0” in Germany, see [Bun14].

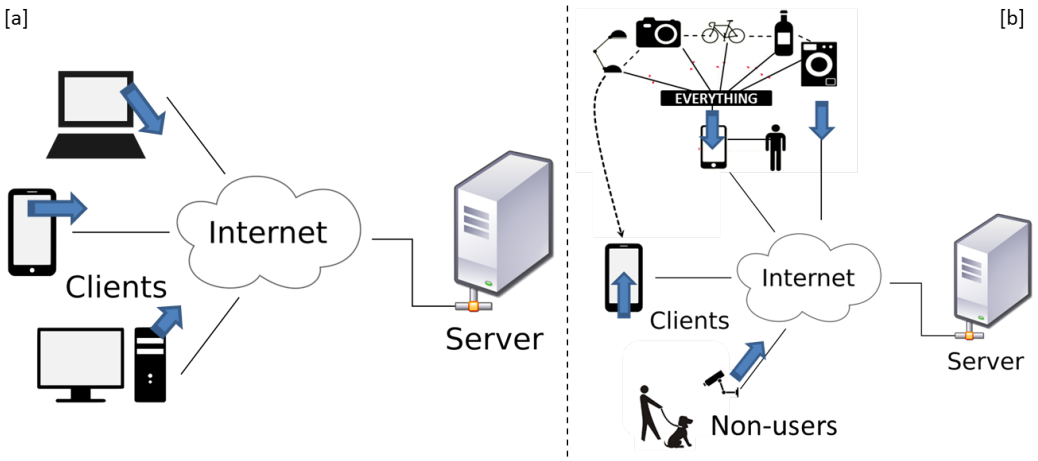

Figure 2.1: Generation of data in [a] web based systems and [b] IoT

Privacy pervasive applications in web based systems operate on the data that has been generated by the users themselves or has been generated according to user activities. IoT based systems extend the structure of web based systems by adding devices around web based clients. The devices do not necessarily interact with the user, they are provided with simple interfaces and communicate in local intranets.

Data generation in IoT follows many automatisms where devices are programmed or follow policies to automatically collect data. These devices use sensors to measure their status and the status of their surroundings. IoT devices tend to change their location and form new intranets dynamically. Consequently, they sense the status of anything that occasionally happens in the devices' environments.

In regards to privacy, this is the most critical difference, as sensitive data of non-users maybe collected and introduced to the system without their knowledge.

2.3 Methodology of Identification of Building Blocks for IoT

In this Section the overall methodology to identify building blocks for the Internet of Things will be described. Specifically, a *"domain model"* is required which conceptually unifies and relates different building blocks.

Carrying out the methodology is a non-trivial task that takes considerable

effort, therefore this thesis will use and extend the results of the projects IoT-A, see Section 2.4.2, and Rerum, see Section 2.4.3.1. The methodology explained in the upcoming Sections follows the approach of IoT-A and Rerum, although it is presented in a simplified and exemplary manner. Additionally, this Section will look into related work and will propose a working definition for the Internet of Things, see Section 2.3.1, that will be used for the rest of this thesis.

The methodology starts with a definition. In general semantics a "definition" can be of two types: intensional and extensional, see [Lyo77]. Intensional definitions describe the essence of a term, while extensional definitions describe element per element how a term is composed. Figure 2.2 displays the methodology.

Figure 2.2: From a definition and building blocks to a common architectural reference

There are many definitions for the Internet of Things which vary depending on the definition's respective point of view. Their nature is intensional, as the Internet of Things has no consensus and cannot be defined extensionally. That means, that a definition can merely frame the Internet of Things, differentiating it from traditional ICT systems and other related technologies (RFID, sensor networks, the world wide web, etc.).

With an intensional definition, applications and scenarios can be identified as scenarios using IoT technologies. These scenarios can be analysed in order to identify common elements. These elements can again be put into an abstract relation and classified as: (1) scenario and technically specific elements or (2) conceptual elements. Conceptual elements that are non-technical and do not change over considerable time are called building blocks. These building blocks and their relations can be summed up in an architectural model, which can be used to develop conceptual extensions for example to support data protection.

2.3.1 Definition

In this Section, a definition is searched that answers the following question:
What building blocks do impact areas described in Section 2.1 have in common?

The question is based on the following idea:
IoT is the underlying technology used to solve the challenges of several application fields. These application fields and use cases have similar problems, which can be conceptual, technical, architectural, etc. Similar problems are solved by similar solutions. This solutions can be abstracted to building blocks. Identifying common building blocks for the application fields is beneficial and can help to maintain interoperability, to build system roadmaps, product life cycles and to benchmark systems, see [BBF+13]. Once these building blocks are understood, they can be analysed from a privacy point of view, used to derive privacy requirements and enhanced to meet those requirements.

Definition 1 - Miorandi et al. define IoT in [MSPC12] *"from a conceptual standpoint [...] IoT builds on three pillars, related to the ability of smart objects to be identifiable (anything identifies itself), to communicate (anything communicates) and to interact (anything interacts). [These objects] either among themselves, [build] networks of interconnected objects, or with end-users or other entities in the network."* Miorandi et al. underline the properties of so called "smart objects" which are physical entities that are able to be identified, to interact with their environment and to communicate with other entities in a wide network.

Definition 2 - Gubbi et al. define IoT in [GBMP13] from an information oriented (Gubbi et al. call it user-oriented) perspective: "*interconnection of sensing and actuating devices providing the ability to share information across platforms through a unified framework, developing a common operating picture for enabling innovative applications. This is achieved by seamless large scale sensing, data analytics and information representation [...]*" Gubbi et al. do not elaborate on devices or the technology underlying the network to "*allow long-lasting applications to be developed and deployed using the available state-of-the-art protocols at any given point in time.*"

Definition 3 - Atzori et al. do not give a definition in [AIM10a], they see the Internet of Things as an intersection of different paradigms, namely:

- the *internet-oriented* paradigm that describes protocols, networks, virtualization through middleware, etc.
- the *things oriented* paradigm that looks at the technology close to physical entities, such as Near Field Communication (NFC), Radio-Frequency Identification (RFID), sensors, actuators, etc.
- and the *semantic oriented* paradigm that helps to create knowledge from information through data analysis.

In summary, Atzori et al. see IoT as a way of using existing paradigms. As a result, the building blocks for IoT will emerge per use case and per application field.

Definition 4 - A possible definition, given by the European Research Cluster on the Internet of Things (IERC) in [VFG+11], was presented in the introduction of this thesis. The IERC states that IoT *"could be defined"* as a "*dynamic global network infrastructure [...] where physical and virtual "things" have identities, physical attributes, and virtual personalities and use intelligent interfaces, and are seamlessly integrated into the information network.*" The IERC envisions physical objects with respective virtual representations that share some attributes. Virtual entities might also be of a more complex character such that they are able to have own identities and interact individually with other entities in a global network.

The four definitions can be summarized to obtain a broader, more complete definition of IoT. This definition will be used as the working, intensional definition of this thesis. The definition reads as follows:

Start of Definition 5. The Internet of Things is a large network that unifies several entities. The majority of these entities are virtual representations (also called virtual entities) of physical entities. Physical entities are measured and changed by small hardware, usually called *smart objects*. A smart object is either hardware (e.g., sensors and actuators) that measures and acts on a physical entity combined with the respective software, or, the composition of the physical entity that is measured and acted on and the respective hardware/software in such a way that they are visually inseparable.

Smart objects can be unequivocally identified, communicate what they sense and they react to to other network participants, taking their environment into account and possibly changing on it.

Virtual representations take the data from smart objects. A virtual representation may take data from one or many smart objects. Complex virtual representations may have complex logics, called *personalities*.

The data that is offered by smart objects to virtual representations is analysed and converted to different data formats depending on their context[9]. Smart objects, virtual representations, data exchange elements, etc. are called building blocks. Building blocks are conceptual elements and are based on different technologies, which may change over time. The change depends on different factors, inter alia the respective state-of-the-art, market position of technology drivers, patents, intellectual property and others. **End of Definition 5.**

At this point it is noted that only partial consensus could be found with Atzori et al. intersection of specifically things/internet/semantics as many other research and application fields may influence IoT as shown in later Chapters of this thesis. Also, some technologies are developed specifically for IoT, therefore not fitting in the intersection paradigm.

In the next subsection, the answer to the question asked at the beginning of this Section will be sketched by looking at two use cases and applying Definition 5 to them. In this process, the building blocks of IoT will be pointed out, set into relation and introduced to an architectural model.

2.3.2 Building Blocks

In the last Section several possible definitions of IoT were presented and a holistic definition for the Internet of Things was proposed. This Section takes a look at building blocks. Building blocks of a system are conceptual, non-technical elements that are not expected to change over the next decades or longer [BBF+13, p. 114]. By looking at two simple use cases and trying to identify their building blocks

[9]For example a user's location data can be transmitted as raw data for user navigation or it needs to be transformed to create a routable road network. Both scenarios may take place when a user is driving his car and the data is measured the same way (by a GPS device), but the data format changes depending on what the user wants (the context of use), as seen in [CK09].

with help of Section's 2.3.1 Definition 5, this Section will try to identify building blocks that answer the question:

What do building blocks that impact areas in IoT have in common?

Figure 2.3: A Smart Home Scenario (left) and a Fitness Tracker Scenario (right)

Figure 2.3 displays two typical scenarios in IoT[10]. The left image shows a smart home. Users that live in smart homes are able to use information technology to control appliances and to adapt their energy consumption (smart homes will be able to produce, store and sell energy) via graphical interfaces such as smart phones or so called energy management systems.

The second image shows one of many scenarios for body area networks and wearable devices. Wearable devices are accesoires that help to measure physical attributes of its possessor. A user can connect one or several accessoires to his smart phone (or other devices with a graphical interface) and analyse his own body data to derive his physical status and change his behaviour accordingly. In the image, a jogger who monitors his heart rate with a fitness tracker is depicted. The user can then adapt his speed to train different energy systems (e.g. phosphagen, aerobic, anaerobic).

Definition 5 will now be mapped to each of the scenarios to identify common building blocks. Definition 5 names five characteristics of IoT: physical entities, their virtual representation, devices that measure or act on physical entities, basic functionalities of the devices and the network structure that interconnects all of these entities.

Smart Home Scenario

Physical entities. The smart home has several physical entities and appliances like heating systems, light bulbs, windows, doors, etc.

[10]Images where acquired under Flickr ⓒ creative commons license.

Virtual representations or virtual entities. According to Definition 5, all of the smart home's physical entities have a virtual representation. These could be software artefacts that are instantiated in the user's smart phone or energy management system.

Devices. According to Definition 5, virtual entities have attributes in order to resemble the status of the physical entity. In this scenario, devices are embedded into the smart appliances.

Functionality of the different Entities. According to Definition 5, entities are identifiable, are able to communicate and to respond to messages from other entities. All of those tasks are carried out by the entity's respective device(s).

Network. Devices and the smart phone or the Energy Management System (EMS) are connected over a home area network. In case third party services are used to analyse the energy consumption, the smart phone or EMS serve as a gateway to the internet.

Fitness Tracker Scenario

Physical entities. In this scenario the user is the physical entity.

Virtual representations or virtual entities. The virtual entity represents the user, with one attribute for his heart rate. This could be a software artefact that is instantiated in the user's smart phone.

Devices. According to Definition 5, virtual entities have attributes in order to resemble the status of the physical entity. In this scenario, the fitness tracker and the smart phone are two devices that are able to measure the user's attributes.

Functionality of the different Entities. According to Definition 5, entities are identifiable, are able to communicate and to respond to messages from other entities. The fitness tracker measures the user's heart rate and provides this data to the user (on his smart phone). The device which monitors the physical entity (i.e. the user) is represented by the fitness tracker while the smart phone holds the virtual entity as a representation of the user.

Network. The fitness tracker is connected to the smart phone, building a small

body area network. In case third party services are used to analyse the user's heart rate, the smart phone serves as a gateway to the internet.

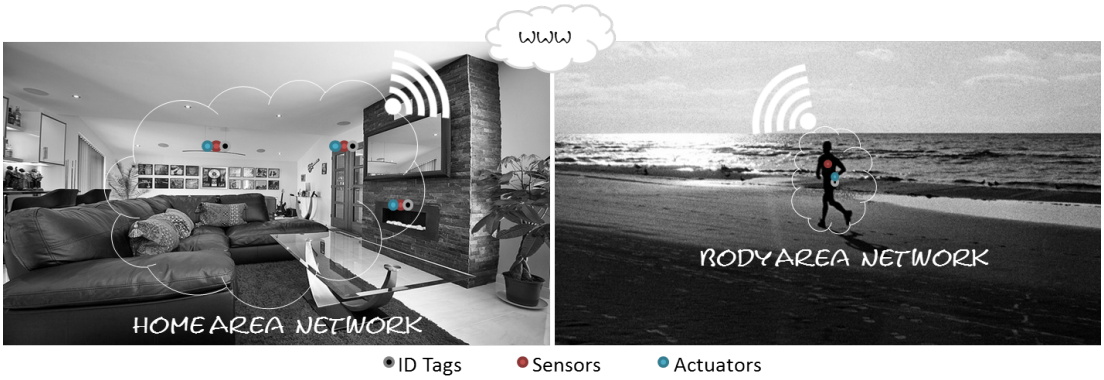

Figure 2.4: The same building blocks could be identified in both scenarios

Figure 2.4 visualizes the identified building blocks per scenario. It is also visible that all building blocks have some kind of relationship to each other: Sensing, actuating and identification elements are in physical proximity. They are either devices themselves or a part of a device that is attached or somehow related to a physical entity. Devices are connected to a gateway, be it the user's smart phone or the energy management system, creating a local IoT intranet with a gateway to the WWW. The smart phone (or the EMS) hosts the virtual representations of the physical entities and acts as a middleware that creates interfaces for graphical applications. After this analysis, the question at the beginning of this Section can be answered with: *the impact areas described in Section 2.1 have at least the five building blocks of Definition 5 in common.*

"At least" underlines that components, their functionalities, resources and services can be put into relation in more exactitude. The result could be an architectural reference model that abstracts the individual building blocks and the architectures from scenarios into general building blocks to facilitate development, standardization and comparability of IoT systems.

Figure 2.5 shows the methodology. This methodology has been described in [Mul08, BBF+13]. The bottom up approach (from use cases to a theoretical

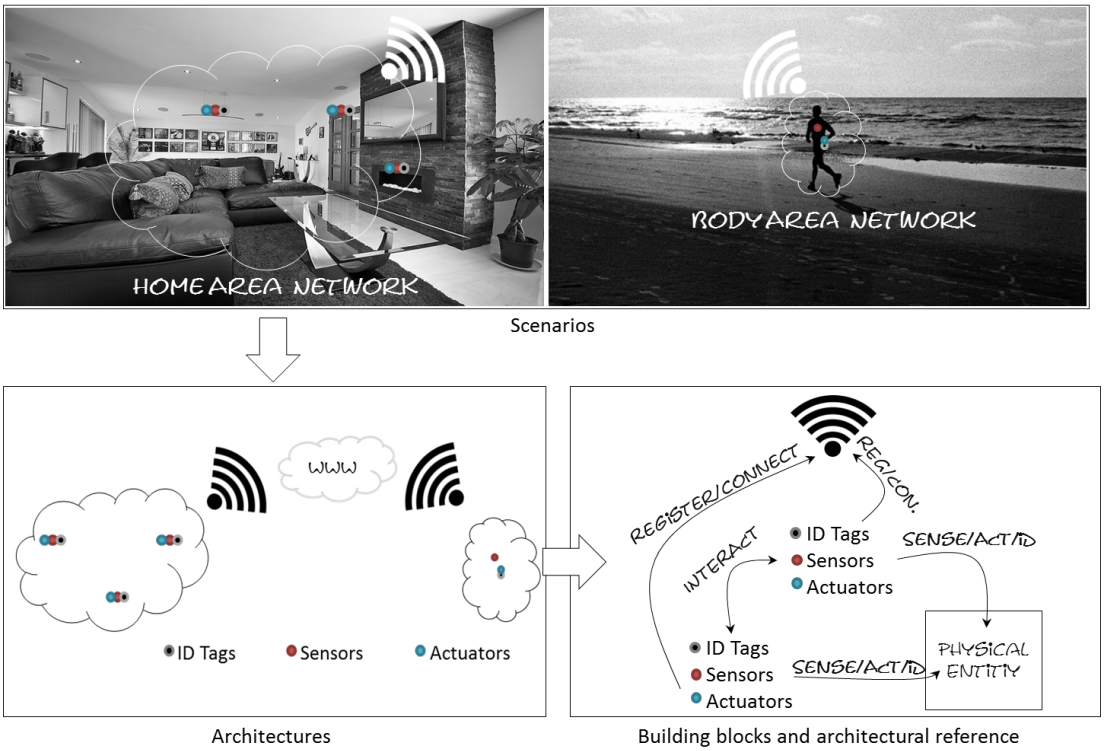

Figure 2.5: From scenarios (top) to architectures (bottom left) to building blocks and an architectural reference model (bottom right)

model) is applied in Internet of Things Archtiecture Project (IoT-A)[11] because there are many IoT systems in use already.

[11]IOT-A, project full title: The Internet-of-Things Architecture, was a FP7 project founded by the European Commission under the grant agreement no.: 257521. http://www.iot-a.eu, last visited August 17, 2017.

An architectural model has to analyse many use cases at once and verify the universal applicability among them. This effort has been done by IoT-A and is therefore out of scope of this thesis.

At the time of this writing, there is no standardized architectural reference model with assigned building blocks for IoT, therefore it cannot be guaranteed that a particular proposal will be long-lived. Nevertheless, the IERC project IoT-A has developed an Architectural Reference Model (ARM) that surpasses the scope of many related projects and that methodologically builds and summarizes its results. How the IoT-A ARM differs from other proposals and how it is build will be thematized in the following Section.

2.4 Architecture Reference and Domain Model for IoT

The IoT-A ARM is a very complete and expedient proposal for a common understanding of IoT in terms of identifying building blocks and relationships in a non-technical manner. Before going into details, general work related to architecture proposals in IoT will be discussed. The discussion will exclude projects that are based on the IoT-ARM such as iCORE, BUTLER, IoT@Work and OPENIoT as they are all based on the same ARM. Rerum, which provides privacy and security extensions to the IoT-ARM, is discussed in Section 2.4.3.1.

2.4.1 Related Work

Gubbi et al. present an "internet" centric IoT architecture which focuses on processes and services more than the "things" centric view of IoT. The exclusion of the "things" centric view raises many questions when mapping the architecture to scenarios as done in this Section. Gubbi et al. give hints on the technology that could be important for smart objects and "things", but it is not explained how these technologies interplay or are related to the other elements of the architecture. For example, RFID is named as an important element of IoT, but it is not explained by whom and for what reason RFID data is processed. Following Definition 5 from Section 2.3.1, it could be assumed that a virtual representation will use that

information, but this is an assumption that lies outside of the architecture. Gubbi et al. reference IoT-A as an *"European Union [project] that [has] been addressing the challenges particularly from a wireless sensor network perspective"*. In the research of IoT-A done for this thesis, this cannot be confirmed. IoT-A's proposal does not focus on particular Wireless Sensor Networks (WSN) related topics like topologies or routing paths, it rather covers a wide spectrum from sensors to business processes, maintaining a non-technical perspective.

Khan et al. [KKZK12] propose a layer based stack, which conceptually characterizes the communication in machine-to-machine environments. It is based on the Open Systems Interconnection model (OSI model), but changes many of the layers in the OSI model. Vidal et al. also propose in [VMZS+13] to change the Open Systems Interconnection model (OSI model) in favour of the host-identity protocol (see [MNJH08]). Although both proposals tackle different problems in IoT, it is uncertain if any proposal that radically changes the OSI model will be accepted and applied by manufacturers, vendors and service providers. It seems therefore that standardization in the IETF for IoT does only consider proposals within the OSI model, such as the Constrained Application Protocol (CoAP) and IPv6 over Low power Wireless Personal Area Network (6lowPAN), as seen in [ICT+13].

Yashiro et al. propose in [YKKS13] an architecture that allows an extension of the CoAP protocol with the uID architecture (see [KS10] for details on the uID architecture). The proposal details technical aspects on the identification of entities in the IoT network and how they can be embedded in applications, but it does not look into relationships or building blocks in general.

There are other proposals that seek to establish an architecture for fast time-to-market services and products. Such an architecture can be found in [FRE15], which discusses several protocols and technologies that are provided by the author's products.

The ISO/IEC JTC 1/WG 10 Working Group on Internet of Things is developing a reference architecture *"to describe the characteristics and aspect of IoT systems, to define the IoT domains, to describe the reference model of IoT systems and to describe interoperability of IoT entities"*, see [San15]. The working group has been

established in 2014 and is still actively developing their proposal for the ISO/IEC 30141 standard. The working group has been in contact with several IERC groups including the Rerum consortium[12]. There is a high probability that the resulting standard will be compatible with the IoT-A ARM and that the concepts and technologies developed for the Rerum project and for this thesis can be applied to the ISO ARM as well.

2.4.2 IoT-A Architectural Reference Model

The following subsection describes the general concepts of the IoT-A ARM. The subsection relies on the description of the IoT-A book, see [BBF+13], if not cited otherwise.

Mueller describes in [Mul08] an architectural reference model as *"captur[ing] the essence of the architecture of a collection of systems"*. Similar to the example methodology of Section 2.3.1, the building blocks of several (existing) systems are analysed and put into relation.

IoT-A classifies the building blocks of IoT within several architectural views. Views are *"representation[s] of one or more structural aspects of an architecture that illustrates how the architecture addresses one or more concerns held by one or more of its stakeholders"*, see [RW12]. Views are composed of viewpoints. Viewpoints are described in [MEH01] as *"establish[ing] the languages or notations used to create a view, the conventions for interpreting it, and any associated analytic techniques that might be used with the view"*.

IoT-A proposes the following architectural views for IoT:

Physical Entity View refers to the physical entities in IoT which may be any physical object that is relevant from a user or application perspective. This can be any physical object including humans and animals.

Deployment View is derived from the physical entity view. For example, the Physical Entity View might indicate that entities are fragile and need to be attached to devices all at once.

[12] See https://ict-rerum.eu/765-2/ for a snapshot of the collaboration. Website was last visited on August 17, 2017.

Operational View describes which tasks and actions are required for operating the system.

IoT Context View depends on the physical entity view, it describes the relationships, dependencies and interactions between the system and its environment.

IoT-A further defines four main activities for the generation of an architecture from the IoT-A ARM, which relate to the points described above. The activities are:

- Create a Physical Entity View. Similar to example of Section 2.3.1, a system is planned by identifying physical objects, their relationships with sensors, actuators and other devices, and the interactions between each other and between components outside the system.
- Create an IoT Context view. This action prepares the elicitation of requirements by further analysing the systems relationships with its environment. IoT-A provides an *IoT Domain Model* which gives an ontological and semantic overview of the possible relationships.
- Elicit Requirements (functional, non-functional).
- Derive other views (deployment and operational).

The last two actions depend on the outcome of the first two actions, which depend themselves on the *IoT Domain Model*. The domain model is introduced in [BBF+13, p. 114] as a "precondition" for working with the IoT-ARM as it introduces the concepts like Devices, IoT Services, Virtual Entities (VE), and their relations. The domain model is comparable to Figure 2.5 where the depicted architectural reference model contains the blocks "tags, sensors, actuators", etc.

The IoT-A domain model defines fundamental building blocks, sets them in relation and serves as a starting point for the creation of the ARM, architectures and IoT systems. The building blocks of the IoT Domain Model are not bound to particular technologies and are not expected to change *"over the next decades or longer"*, see [BBF+13, p. 114].

In the next Section, the IoT Domain Model will be explained followed by the extension of the domain for security and privacy building blocks in Section 2.4.3.1.

2.4.3 Domain Model

The IoT Domain Model provides a common definition and taxonomy of the IoT domain. The complete IoT-A taxonomy for the Internet of Things is given in [BBF+13, Section 6.7]. The domain model consists of several sub-models, namely the IoT Information Model, the IoT Functional Model, the IoT Communication Model and the IoT Trust, Security & Privacy Model.

The IoT Information Model defines the structure of information in an IoT system without giving details on its exact representation. The IoT Functional Model identifies functionality groups that manage the information of the Information Model. The Communication Model describes concepts for handling the complexity of communication in IoT, while the Trust, Security and Privacy Model introduces typical functionalities and interdependencies for trust, security and privacy.

IoT-A provides a graphical representation of the domain model using the UML language.

Figure 2.6 shows the diagram. The diagram sets several blocks into relation. Each block has a colour, yellow depicts animate objects (humans, animals, plants and so on), blue is considered hardware, green is considered software and white is not clearly classifiable (e.g. a scenario specific, non-statical combination). The diagram evolves around the physical entity. A physical entity may be an animate object, but also any kind of physical ("touchable") object. Physical objects maybe atomic or composed of many physical objects, such as the subcomponents in a car. As seen in the examples of Section 2.3.1, physical objects are measured and acted on. Therefore, one or many devices are attached on physical entities either physically or in its physical proximity. A device, as a physical ("touchable") object, is itself a physical entity and could be measured, e.g., for maintenance. Devices may have different duties but three basic functionalities can be observed for IoT (see Definition 5 in Section 2.3.1): identify, measure and act on entities. These functionalities are carried out by devices (tags, sensors and actuators) which can be individual devices, or, one device may carry out all functions. A particular relationship may exist, where a sensing device is also measuring the identification

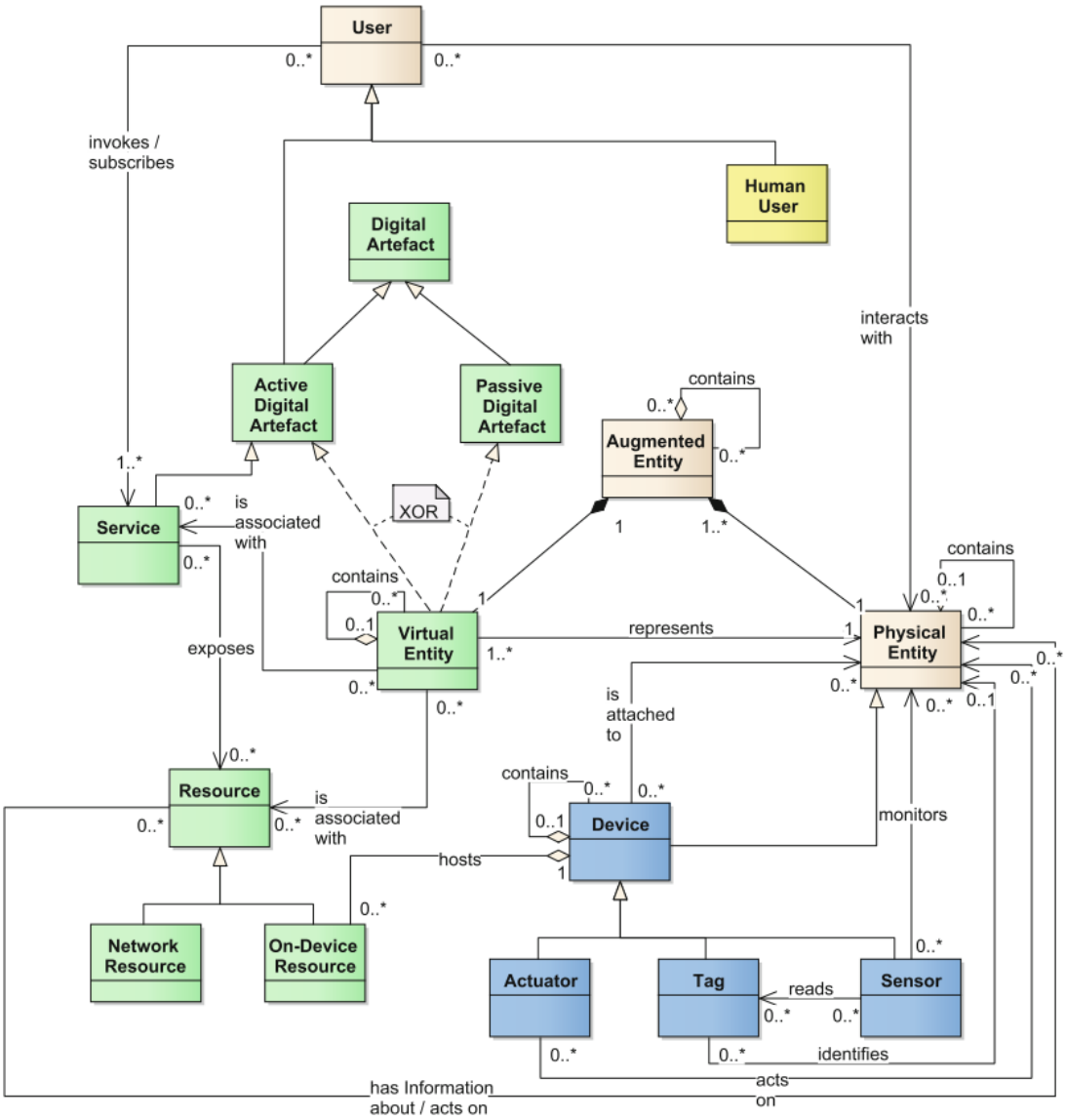

Figure 2.6: The IoT-A: internet of things domain model

device. In general, these functionalities are not mandatory, therefore their relation to the physical entity is zero to many.

In Definition 5, the idea of virtual representation of physical objects was already introduced. This idea is also found in the domain model with the relation of physical to virtual entities. The relationship is denoted with [one to many] to [one], as there may not be a Virtual Entity (VE) without a Physical Entity (PE) that the VE is representing. Also, a physical entity may have several virtual representations that are either atomic or complex, but a VE is always related to a single PE. The existing relation of PE and VE generates a conceptual entity called Augmented Entity (AE). The AE is the composition of a PE and a VE. If one of the elements is removed, the AE is destroyed.

Each VE has an Identifier (ID) that identifies it unequivocally. A VE can be either passive or active. Active VEs are of the type Active Digital Artefacts (ADA) such as software applications (Services are also ADAs, see below). Passive types are Passive Digital Artefacts can be static software objects such as database entries.

The VE can be imagined as a software object that is initiated with the attributes of its physical counterpart. For example, if a window closes, the VE of that window has to change one of its attributes to 'closed'.

In the domain model, the VE does not directly consume sensed information from a sensor. The reason is practical: a sensor is likely to have proprietary protocols and interfaces. Therefore an abstraction of the data interface has to be found. IoT-A proposes to call those abstraction resources. Accordingly, every device hosts two types of resources, a network resource that takes care of communication functionality (receiving and sending messages) and an on-device (data) resource, that handles the interfaces of providing and receiving (and understanding) data.

A VE is associated to zero or many resources and is accessed through services. Services (or agents) are software clients that provide interfaces to resources and VE. In the case of resources, services offer interfaces to retrieve or provide information. In the case of VEs, the service's interfaces allow to change a VE ("close window") and to trigger a device reaction accordingly ("actuator closes window").

Users are animate objects or some kind of a Digital Artefact (e.g., a Service or an application) that interact with Physical Entities. In the case of direct physical interactions (e.g. opening or closing a window), the VE has to adapt its attributes. The same interaction can only happen virtually, if the PE can be acted on by an actuator. It is then the responsibility of the Services to check if such an interaction is possible.

The domain model will serve as the fundamental building block for the rest of this thesis. It will be extended throughout this thesis and discussed in Sections 2.4.3.1.

Considerations The domain model suggests that VEs do not need to be associated to any service or resource. An example given in [BBF+13] is a passive VE in form of a database entry. But if the VE is not associated with any service or resource, it is not able to represent the physical entity adequately. If the PE changes, there is no other way of changing the VE than updating the database. The database has to be an active digital artefact that needs a service to be updated. How and when the service updates the database is uncertain, as there is no resource (and device) to trigger the alert. With improper representation, the composition of the Augmented Entity cannot be sustained and the PE does consequently not participate in the system.

If a human user was to update the database (assuming interaction also means monitoring), he has to invoke a service to update the database. The database would then contain the VE that is changed by the service. In this case, the VE had to be explicitly contained in the database, which is not a requirement following the zero to one, zero to zero self-relationship of Virtual Entities in the domain model.

Therefore, the assumption will be made that Passive Virtual Entities which are neither associated to devices nor services, are always contained by an Active Virtual Entity with both associations. This assumption[13] will be included in the final domain model of this thesis in Section 2.5.3.5.

[13]This aspect is newly introduced in this thesis. The BUTLER project has introduced an additional building block called Context that is related to VEs, but it does not address this aspect directly. The final domain model of this thesis will consider the work from BUTLER, but the outcome is a proposal firstly published in this thesis

2.4.3.1 Rerum Domain Model

In this Section, the first phase Rerum domain model, see [RER15], is introduced. In the first phase, functional and non-functional requirements of Rerum use cases have been introduced to extend the IoT-A domain model.

Rerum First Phase Domain Model. Rerum has based its domain model on the proposals of the IoT-A and BUTLER projects. Figure 2.7 is taken directly from deliverable D2.3 [RER14b] and presents the first phase domain model. In comparison to the IoT-A domain model, see Figure 2.6, the Rerum domain model adds six new elements:

Context. The notion has been adapted from the BUTLER project. In short, context is an additional attribute of a Virtual Entity, that describes the context of the Physical Entity. Contextual information can be very different: e.g., from personal information like position to information about the state of an entity, like temperature and light level.

Context has a zero to many relationship with Resources, as contextual information maybe measured from none or many Devices that are or are not related to the Physical Entity the Context is describing. Context is exposed by its Virtual Entity, as it can be described as an additional property that the VE carries. Context is accessed by a Service (thus zero to many relationship), as is any other attribute of the VE.

Users and Applications. Rerum assumes that Software invokes Services, thus replacing the Active Digital Artefact with Applications. Rerum also assumes that a Human User either interacts physically with an Entity or uses an Application to invoke Services and interact with an Entity's virtual representation. The associations are zero to many in both cases (user to application and application to service). A self-association is depicted for Services to allow one Service to invoke another. This can be useful in case of federations, collaborations, compositions, etc.

Virtual (Rerum) Device. The Virtual Rerum Device (VRD)[14] is the virtual representation of a Device. The VRD behaves like the VE of the IoT-A

[14]Rerum branded this entity with the Rerum label as the project is developing devices itself.

domain model: it can be self-contained (a VRD can contain zero to many other VRD's and accordingly be associated to none or one VRD head in case of a composition). A Device is assigned to a PE, therefore the VRD is assigned to a VE as well. The associations are zero to many.

Generic Virtual Objects. Rerum sees VRDs and VEs as similar objects that represent different entities. Rerum therefore makes an abstraction of both and defines an objects called the Generic Virtual Object (GVO).

Administrator. Rerum defines an Administrator as a human user with the highest privileges in the domain model. He administrates the software that contains virtual representations, is responsible for associating virtual entities with their physical counterparts, registering new devices and keeping the firmware of devices up-to-date.

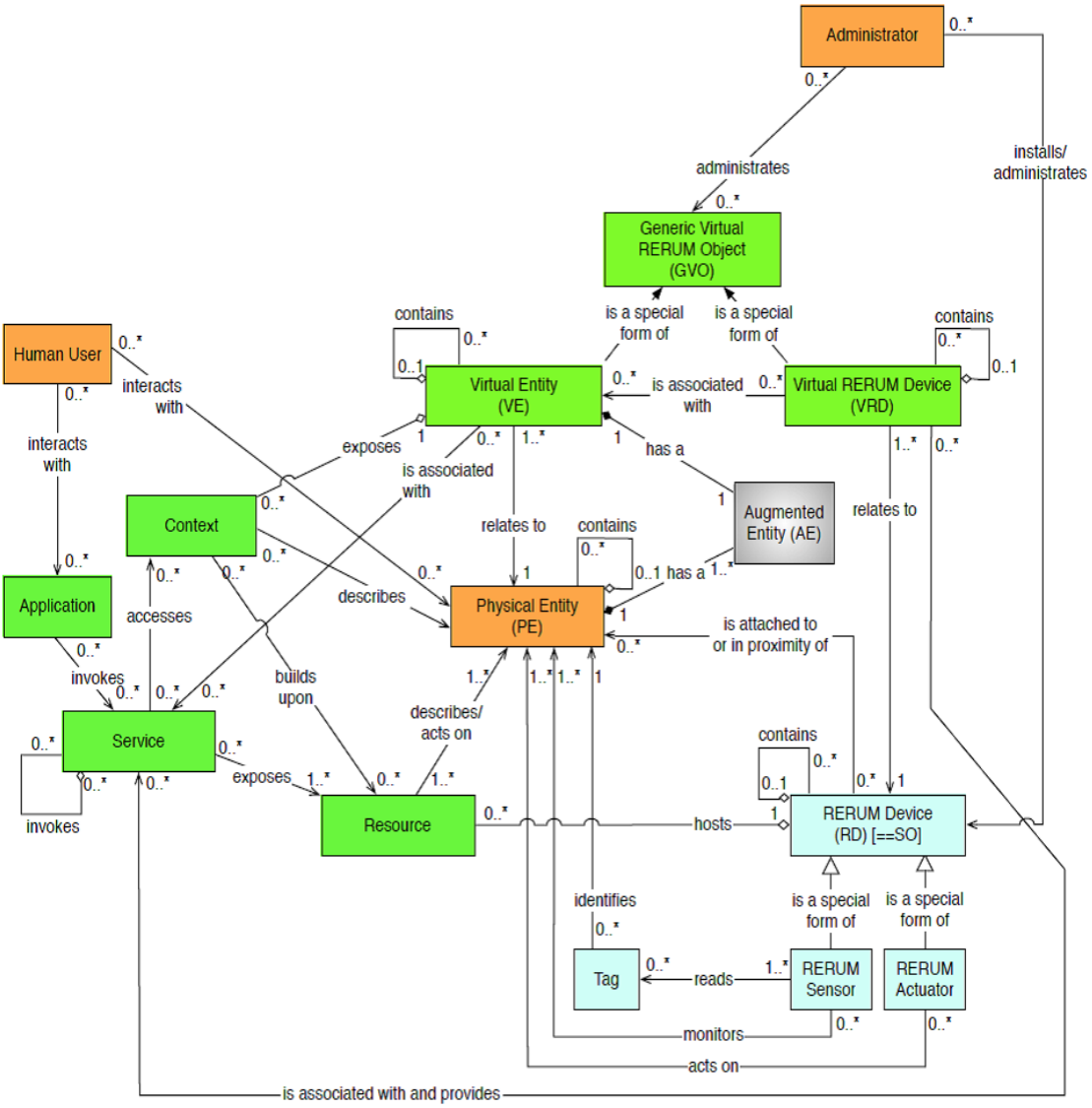

Figure 2.7: Rerum first phase ARM

In the second phase, privacy, security and trust technologies were developed. As these technologies are used in the whole architecture, a further extension is needed, which leads to the final Rerum domain model. Several of the privacy enhancing technologies that were developed for this thesis are among those that influenced the domain model. Figure 2.8 gives an overview.

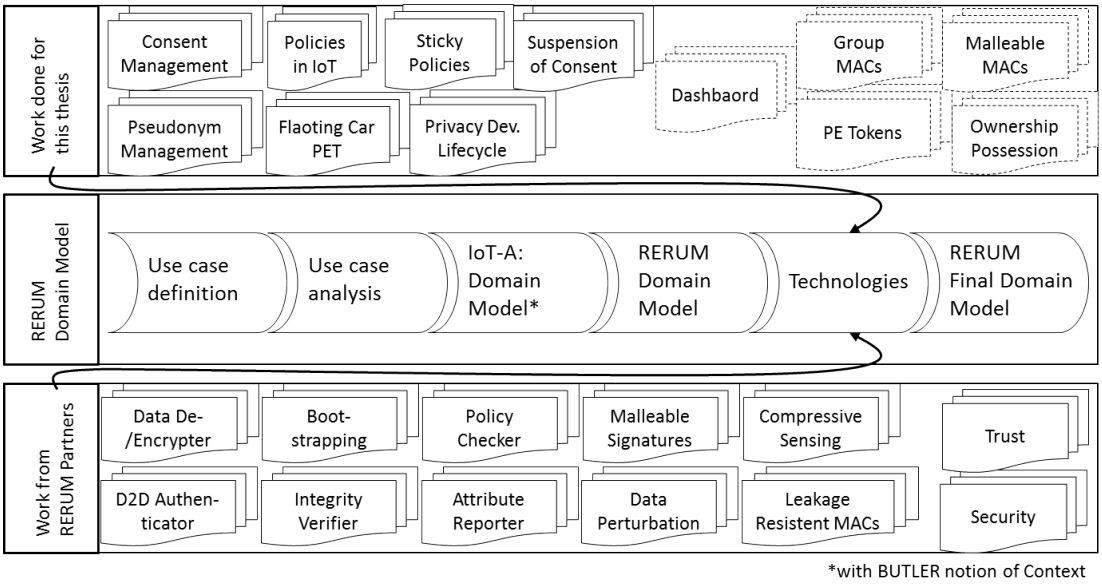

Figure 2.8: Contributions for Privacy Enhancing Technologies in Rerum

The depicted technologies are presented in detail in Chapter 4.

This thesis builds upon the first phase and the final domain model to propose a privacy enhanced domain model[15] in Chapter 5.

2.5 Further Architectural Concepts

Rerum defines additional layers to complement the domain model. The domain model defines which elements and relations a system has while the layers describe how they interact. In total, four layers are defined:

On-device layer. This layer describes the software, hardware components of devices and also handles how technologies can be embedded into devices.

Communication layer. This layer describes how active entities of the domain interact with one another. The communications scheme is significant for the analysis of threads and risks as well as for the selection of protection mechanisms.

Middleware layer. This layer manages the virtualization of entities, e.g., creating a virtual entity from a physical entity and associating nearby devices to that virtual entity. Other tasks are the registration of new devices, the

[15]The concepts that are additionally presented in this thesis could not be integrated in the Rerum project due to the timely finalization of the domain model. In any case, the Rerum domain model can bee seen as a living proposal that is meant to be used, adapted and extended.

orchestration of service- and discovery requests, how services and resources are provided as well as the association of different entities according to Rerum's IoT domain model.

Application layer. This layer is an abstraction layer that specifies the protocols and interfaces that interact with all the underlying layers. For example, the business process modelling language is used is this layer. Rerum has defined this layer as out of scope.

Rerum's exclusion of the application layer has also an effect on the layout and scope of this thesis, as architectural work and the verification on trials were not available. The application layer deals with the user interaction and user experience which is a central part of *privacy as control*. Therefore some details and suggestions are given in Chapter 4.

In the following subsections, the communication layer is further discussed as its functionality is significant for the data flow of an IoT system and for the privacy analysis of that system. The rest of the layers will not be further discussed here as the application layer has been left out of scope. Rerum's on-device layer as well as the middleware layer have already been discussed extensively, see [RER15].

2.5.1 Federation, Collaboration, Cooperation and Composition

In IoT entities interact heavily. Virtual entities communicate, interact and group with other entities to provide services of higher complexity. In this Section[16], group interactions will be discussed. Grouping is a certain behaviour of entities to share or delegate logic to one or among several entities. Rerum describes one form of grouping as forming a *federation*.

Rerum defines in [RER15] federations as follows[17]: *"a grouping of virtual devices is called a "federation" if they cooperate to offer a composed Service for monitoring or controlling a PE. Furthermore, the VRD federation can be considered as a service composition between the Services of the VRDs for accessing/changing the attributes of a VE."*

[16]This Section is firstly published in this thesis.

[17]For legibility, the Rerum terminology has been adapted to the terminology of Section 2.4.2.

The definition uses several terms for the interaction of entities when they "group": "federation", "cooperation" and "composition". This terms can again be differentiated based on how entities associate to one another, if resources of entities are shared or not between other entities and how a common goal, for example the provision of a mutual service, is reached.

In this Section four forms of "grouping" will be reviewed and set into an IoT context, namely *federation, collaboration, cooperation and composition.*

Federation. Serrano et al. define in [SMH+10] federations for future internet systems as: *"any governance environment in which two or more autonomous administrative domains must interoperate for their mutual advantage and where they must establish business, trust and technical agreements to make that happen."* This definition can be applied to IoT terms in the following way:
a federation is a relationship in which entities interact to achieve a mutual goal, e.g., the provision of a complex service where entities are able to operate on their own ("autonomous") and have several associations to other entities and contexts ("administrative domains"). Agreements have to be established to reach the federation's goal, e.g., an agreement on which service the federated entities guarantee. Also technical agreements, e.g. modes of operation, such as how data object types that are used between entities are to be handled, protocols (such as TCP or UDP), interfaces (APIs which describe parameters and function calls), encryption modes and the definition of policies and trust relationships. Trust is essential for federations as one entity has to trust a second entity to behave as expected[18]. In federations, participating entities will not gain access to the resources of other federation entities. Rather, entities manage their resources on their own and they offer individual services.

Serrano et al. define a federation as a governance. Rerum assumes that there is one entity that leads the governance and at the same time offers the service which is also responsible for the logic necessary to orchestrate the federation. In this thesis, the leading entity will be called a federation head, the remaining participants will be called federation peers.

[18]The notion and the mechanisms of trust will be further explained in Section 4

Collaboration. The term collaboration in ICT is not used as a mechanism but rather as a form of "working together" between researchers and system designers to drive innovation or education, see for example [McC04, HW04]. Therefore, a closer look is taken at social sciences, which looks deeper into the interaction in collaboration. The definition of Mattessich et al. in [MM92] is adopted, which describes collaboration as a: *"mutually beneficial and well-defined relationship entered into by two or more organisations to achieve common goals. The relationship includes a commitment to: a definition of mutual relationships and goals; a jointly developed structure and shared responsibility; mutual authority and accountability for success; and sharing of resources and rewards. [...] The individuals who represent collaborating organisations are referred to as partners or members."* Compared to a federation, entities in a collaboration share their resources, they equally define goals, business, trust and technical agreements. A collaboration has no head, it is led in a mutual way by all members. Entities in a collaboration may exclude the provision of services outside of the collaboration, as shared resources may be used and blocked by fellow collaborators.

Cooperation. Cooperation has not been sufficiently discussed in the field of IoT or ICT. For example, Schaffers et al. discuss in [SKP+11] "cooperated federations", but do not give any information on what a cooperation nor what a federation means. Another example can be found in [RPP11], Rohokale et al. present a model for cooperative IoT, but assume that the reader is savvy of wireless cooperation networks and is able to transfer his knowledge to IoT. Sirinivasan et al. refer to game theory to explain cooperative behaviour in their work on cooperation in wireless networks, see [SNCR03].

Game theory will also be used here to find a suitable definition for cooperation. Game theory is a mathematical model to describe social behaviour, specially cooperative and conflictive aspects thereof, and is thus well fitted for explaining the term cooperation.

Dugatkin et al. define in their work on game theory and animal behaviour, see [DR98], a cooperation as: *"an outcome that - despite potential costs to individuals - is "good" (measured by some appropriate fitness measure) for the*

members of a group of two or more individuals and whose achievement requires some sort of collective action. [...] [To] cooperate can mean either to achieve that cooperation (something manifests at group level) or to behave cooperatively - that is, to behave in a manner making cooperation possible (something the individual does), despite the fact that the cooperation will not actually be realized unless other group members have also behaved cooperatively." Furthermore, they describe three paths to achieve cooperation: by reciprocity (i.e. acting in a way that benefits others), by forming a group and through by-product mutualism (an action that first and foremost benefits the actor, but also benefits others as a by-product).

This definition can again be described in IoT terms as follows: entities in a cooperation have a common "outcome" or goal, e.g., the result of some sort of service provision. The goal is mutually defined and benefits all of the cooperating entities. To reach their goal, entities behave in one of the three formulated ways (reciprocal, group oriented, creating by-products). For example, a group of entities in an IoT space would cooperate and act reciprocal, if they chose every action in a way that it benefits a part of the group. In comparison, a group oriented behaviour means that some of the cooperating entities form an explicit group and that their actions do not affect the other entities in the space directly. Lastly, the entities may offer individual services, but they chose to act in such a way that a by-product benefits some of the space's participants.

Entities in a cooperation may share their resources. They are likely to be able to offer individual services anytime. A cooperation has no single point of leadership and may be terminated as soon as the outcome is reached.

Composition. Composition relationships are known in software engineering as part of the unified modelling language. The UML version 1.3 specification, see [RHCF05], describes compositions as: *"a form of aggregation with strong ownership and coincident lifetime of part with the whole. The multiplicity of the aggregate end may not exceed one (it is unshared)."* Entities entering a composition are created for the composition only and are destroyed when the composition's lifetime ends. Composition entities share their resources and are not allowed to share or use them outside of the composition. The notion of ownership suggests

that one entity in the composition has more responsibility than the others. It carries the logic to orchestrate the compositions activities, manages the shared resources, defines the composition's goals and handles service, trust and technical agreements. Clarke describes in [Cla02] additional agreements and actions that need to be made when entities enter a composition relationship. These are adapted to IoT terms:

Specify how entities should be integrated. Define how different entities can be integrated and which integration method is preferred. Methods could be for example migration, replacement or wrapping.

Identify and specify overlaps. Entities might have similar APIs, assignments to devices and other entities, services and resources. These overlaps have to be identified.

Specify how conflicts in corresponding elements are reconciled. If conflicts appear, define how they can be resolved. For example for an API, the API of entity may take precedence over another, or neither is preferred and a new API is created.

2.5.2 Communication Schemes in IoT

Section 2.4 introduced the methodology to obtain an IoT domain model which comprises IoT entities and their relationships. Section 2.5.1 described how entities can group and share common logic. Communication schemes describe how entities communicate. Depending on the scenario, entities might use technology to share messages to entities nearby, they might forward messages through gateways as they lack resources to connect to the WWW, or they might be powerful enough to communicate to entities nearby and to send messages over the WWW to service providers. In this Section four different communication schemes will be presented: cloud based, gateway based, intranet based and distributed communication scheme.

Cloud based communication scheme. Figure 2.9 exemplifies the scheme with two virtual entities, a user device with a graphical interface and two service providers. This scheme does not foresee entities talking to each other. Entities send data to few central service providers over the WWW. These service providers

are typically called cloud services. Cloud services are no entities in the IoT domain, as they do not represent neither a physical entity nor a device. In modern cloud

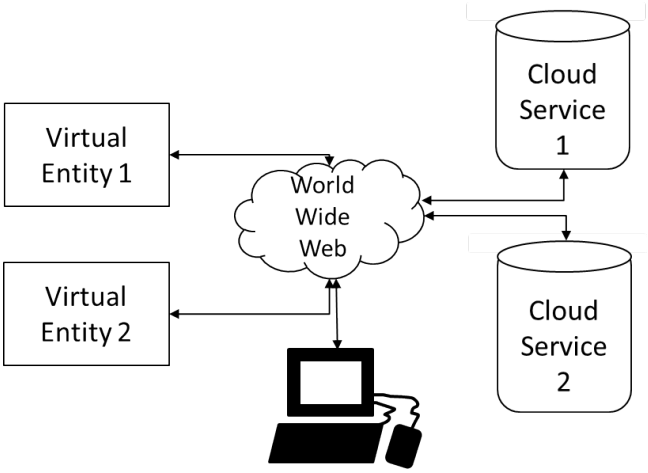

Figure 2.9: Cloud based communication scheme

based communication schemes, cloud services may be defined more loosely and can be part of IoT entities. Also, entities might talk to each other before sending the common data to the cloud service.

Gateway based communication scheme. In IoT, constrained devices may be used to retrieve data and to provide them to the respective virtual entities. Constraints may be in computational power, in battery capacity and in network range. Additionally, constrained devices might be changing their location and they might be unavailable for a long period of time. In order for a cloud based scheme to work with constrained devices, at least one unconstrained entity is needed which can expand the limited network range. Rerum has determined this entity as a gateway, see [RER15]. Figure 2.10 depicts the scheme with the same setup as above. Gateways are predominately being used in sensor networks. Ishaq et al. refer in [ICT+13] to two main forms of gateways: translators of web protocols into proprietary device network protocols for real-time access of data and databases that store data from the device networks without allowing any interaction with the sensors.

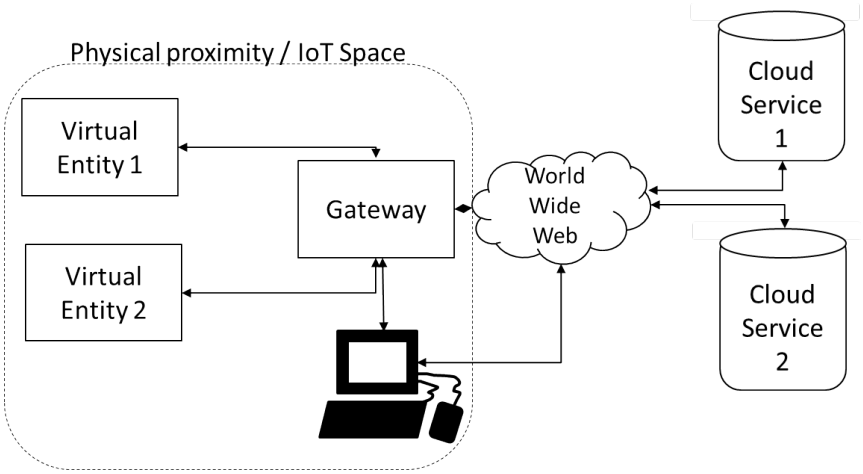

Figure 2.10: Gateway based communication scheme

Intranet based communication scheme. The intranet based communication scheme extends the gateway based scheme by creating a physical proximity view. It allows all nearby entities and devices, also constrained ones, to exchange messages. The physical proximity is also called a *smart space*, see for example [NSFB15, KBG13]. Figure 2.11 again exemplifies the scheme. In this scenario every entity

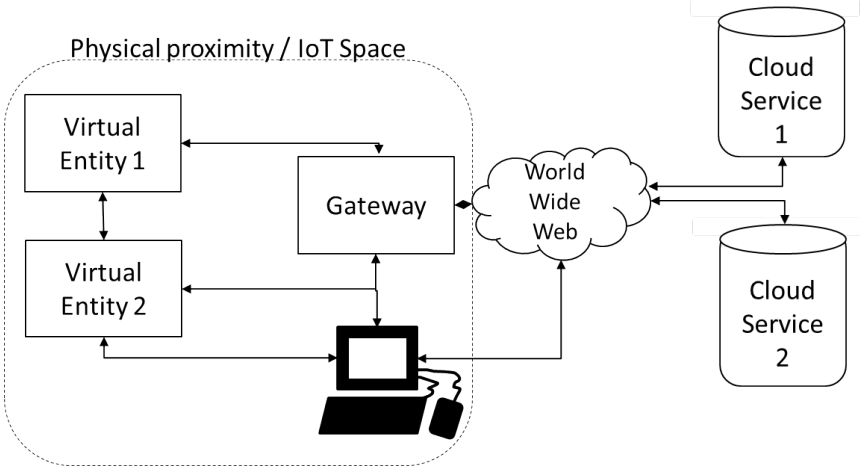

Figure 2.11: Intranet based communication scheme

in the physical proximity is connected, building an intranet. A user can access the data of the VEs with his computer by sending a request directly to the entities.

Distributed communication scheme. The distributed scheme interconnects everything and is the closest to IoT, see definition of Section 2.3.1. As seen in Figure 2.12, every entity, device and service can freely connect to one another without any restriction. The accessibility of every entity to the WWW shows

that all entities in this scheme are either unconstrained or so well organized that messages can be routed through nearby entities until they reach WWW gateways.

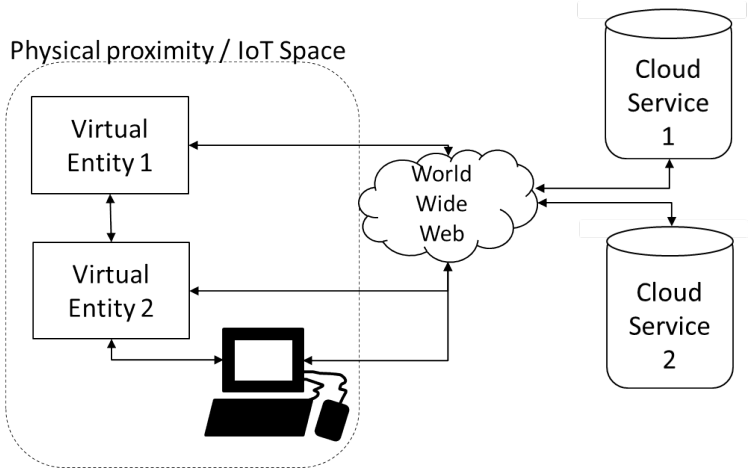

Figure 2.12: Distributed communication scheme

Although the distributed communication scheme is the closest to the definition of IoT, the intranet based scheme seems to be the most adequate to be used in current IoT systems, due to the restrictions of many IoT devices. Ishaq et al. note in [ICT+13] that gateways are common in sensor networks, as they have been used by proprietary vendors to allow a controlled connection between the sensor and actuator networks and the WWW. Gateways have been used predominantly as middlewares, translating web protocols into proprietary sensor protocols. Roman et al. survey in [RZL13] communication and logical schemes and note that the "intranet of things" is the result of the interconnection of current, isolated ICT systems. One example for this is found in smart cities, where smart buildings, smart homes, water supply and energy supply management systems of an entire city are connected to each other, forming a municipality network (the city intranet). Rerum uses a variant of the intranet based communication scheme for several different smart city scenarios as well. Rerum utilizes the gateway as a trusted component which carries some of the Rerum middleware logic. This logic takes care of various tasks like device discovery, device registration, entity management and service publishing. The interested reader is referred to [RER15, Section 4.2.1].

2.5.3 Ownership and Social IoT

Definition 5 in Section 2.3.1 describes the interaction between different smart objects by exchanging data, reacting to other smart objects and their environment. Atzori et al. elaborate on the interaction of smart objects and human users in [AIM14]. Atzori et al. envision a network of objects with social behaviour. This idea matches the description of "personalities" in definition 5, where objects establish trust relationships and act accordingly in the network. Human users are able to interact physically and impose rules on *their* smart objects at the same time. The relationship between a smart object and the human user is not specified, as e.g. in [AIM14], but it is a deciding factor in the idea of social IoT as relationships between objects may be influenced by trust relationships between human users. Best to the author's knowledge, there is no existing research[19] that discusses the idea of the digital representation of real-life ownership relationships. An *ownership model* digitally represents a real life "thing"-to-user relationship in a differentiated, simplified and pragmatic way, where a stakeholder can be a human being, a machine or a virtual entity. Differentiated means, that the model serves an explicit purpose in the Internet of Things domain. It differentiates itself from other mechanisms such as authentication[20]. The property of simplicity takes away complexity from the model, which can be found in its real world counterpart, but is not needed for the model's purpose. One example is the physical contact with a device. Physical possession is important for ownership purposes, but the model does not profit from real physical contact (e.g., a person holding a smart phone in his hands versus just being near the device). Naively said, it is enough to assume that communication in a near field equals physical possession. Finally, the concept of pragmatism describes the generality of the model. The model is not uniquely related to one real word example. It generalizes properties (e.g., an ownership model is always based on a subject *owner* and an object *thing* and allows heterogeneous applicability among similar relationships.

Three crucial concepts define ownership: absolute ownership, physical possession and rights of use. They are found in cross research domains such as

[19] Such as in the discussed conceptual works of [MSPC12, AIM10a, GKN+11, MS08, WBC+09].

[20] Ownership and authentication share similar concepts and technologies, but they serve different and particular purposes, respectively. Further details are discussed below.

jurisprudence and economics, where they constitute the evidence of ownership, i.e., which party has certain rights and duties over a property, how protection of ownership is defined and how ownership can be represented and enforced towards other parties. Ownership models also represent machine to machine and virtual ownership relationships, which are not common in real life scenarios.

2.5.3.1 Ownership Representation in IoT

In real life, ownership is deeply rooted in society, although not actively enforced.

In the Internet of Things, where objects are interconnected and communication is digital and transparent, different ownership enforcement mechanisms are needed. Policies in particular will enforce the rights of use (who is allowed to use a certain "thing" in a certain way?) and to persevere privacy aspects (the owner of a smart object authenticates a subject and authorizes him to use the object and, at the same time, all identities may remain unknown). The owner of an IoT object will decide over the following fundamental attributes:

- The number of identities and pseudonyms.
- The mapping of identities and pseudonyms.
- In case users conflict in the interaction with an IoT object, the owner is the deciding party.
- Closely related to dissolving conflicts, the owner defines a set of fundamental policies that decide over the access permissions to the smart object.

In contrast, the physical possessor of a smart object may, if the fundamental set of policies allow it, overrule some of the owner's attributes:

- The possessor may change the number of identities and pseudonyms in order to hide his interactions (also towards the owner).
- The possessor may be the only person to map new identities and pseudonyms to objects.
- In case users conflict in the interaction with an IoT object, the possessor may be the deciding party. But ultimately the owner holds the final decision, as he holds legal rights over the object.
- The possessor may override some of the owner's policies, if they are related to the protection of the possessor's personal data.

The ownership relation helps to ease the usability of access control. An object may have standard policies set by the owner, but needs to change its policy set in case a person different from the owner is allowed to interact with it. The case of changing from owner to physical possessor is a part of smart object's self-awareness capabilities that benefit the ownership concept as described in the following Section.

2.5.3.2 Related Work

IoT objects are envisioned to be supportive in relation to system architecture, design and development, integrated management, business models and human involvement on their own, and to integrate themselves in any context they are carried into [SU05]. This again is only possible, if objects are capable of being self-aware, i.e., to understand in which context they are in and how they are supposed to behave. Kortuem et al. define in [KKFS10] three types of self-awareness for smart objects: activity-awareness, policy-awareness and process-awareness. In [SU05] and [SS11] awareness is further enhanced with the concepts of context-awareness and self-description semantics.

Activity-aware smart objects are aware of event and activity streams. This objects can react according to their state, to which functionality they are providing, and how they are being used. Typical events are picking up, turning on or operating the object. One example could be a container filled with a chemical reagent. If the container exceeds a certain temperature or pressure, it will disable its opening mechanism.

Policy-aware smart objects understand to what extent their functionality or state comply with predefined policies. In a smart home this could mean that an oven should not turn itself off when heating, even if energy prices are high.

Process-aware smart objects understand the organizational workflow they are part of. They can relate the physical occurrences to their workflow and react accordingly.

Context-aware smart objects share the properties of policy-aware smart objects. They can initiate and form ad-hoc workflows based on semantic description of their surroundings. Several unknown smart objects group together and agree on a

workflow and grouping scheme to achieve a certain result [ZGW05]. Ownership can be regarded as an addition to the awareness property as well. An object can identify who it belongs to and who it is interacting with, enforce access control and draw conclusions in order to build trust relationships.

2.5.3.3 A User-centric Implementation for Ownership

For a clear ownership definition, the German civil code will be used. As existing research shows [Qui11], historical reasons and steady development in *real property*, *personal property* and the differentiation between "Law" and "Equity" have made it difficult to find a clear definition in English Law[21].
Ownership, as defined in the German civil code *§903 Bürgerliches Gesetzbuch*, *"is the right to use a* thing *and to exclude others from any interaction"*. Notably, using this definition, several key concepts from IoT-A and Rerum become visible: the "thing" which corresponds to a Physical Entity in IoT and several forms of "interaction". The article further defines a *positive content* and a *negative content*, which an owner can define regarding his property. The positive content describes which kind of interaction is allowed and the negative content describes which kind of exclusions exist for a "thing". According to German Law, ownership is an absolute right over a thing, as it can be represented and enforced *erga omnes*, i.e., towards all parties.

This definition, even if unknown to many, is ubiquitously present in everyday life. The physical interaction with a thing is coined by the ownership relation of it. If the thing is the property of a person A, a person B will (normally) avoid certain actions with it, if not avoid an interaction at all.

There are of course several kinds of ownership: sole ownership, simple joint ownership and partial joint ownership. All of them describe different relationships of a physical entity to one or many owners. But all of them have in common, that positive and negative content can not be enforced physically any time, anywhere. This is different in the context of IoT.

In order to move towards a complete ownership model, following steps are necessary. First, the ownership relation of things and users must be defined.

[21] English Law means hereby the jurisdiction and legal system of England and Wales. Other jurisdictions, such as those from member states of the Common Law, are not considered.

Second, adequate technologies have to be identified. Here the Rerum architecture and the scheme of *connected intranet of things*[22] are considered.

2.5.3.4 Proposal on the Establishment of Ownership

The following proposal follows a conceptual as well as a technical description of a possible ownership establishment process and gives readers a more concrete use case on how ownership could be used. The Section makes use of an example to guide through the establishment phases. Subsequently, ownership will be integrated in the IoT domian model in the next Section.

The Setup. A user is surrounded by two smart objects. The smart objects could be two wearable devices, such as a fitness tracker and a smart watch. The user additionally has a smart phone[23]. Figure 2.13 demonstrates the setup and denotes some possible examples of adequate enabling technologies.

As described in the *connected intranet of things* scheme, the two physical entities are not able to connect to the world wide web, therefore they are neither searchable nor identifiable over a wider network.

[22] As discussed in Section 2.5.2, this model describes a network similar to a private area network, were smart objects build an intranet. In this intranet only a few objects are able to connect to a wider network, e.g., the world wide web.

[23] Note: all of the objects are physical entities including its respective devices in a way that is visually inseparable.

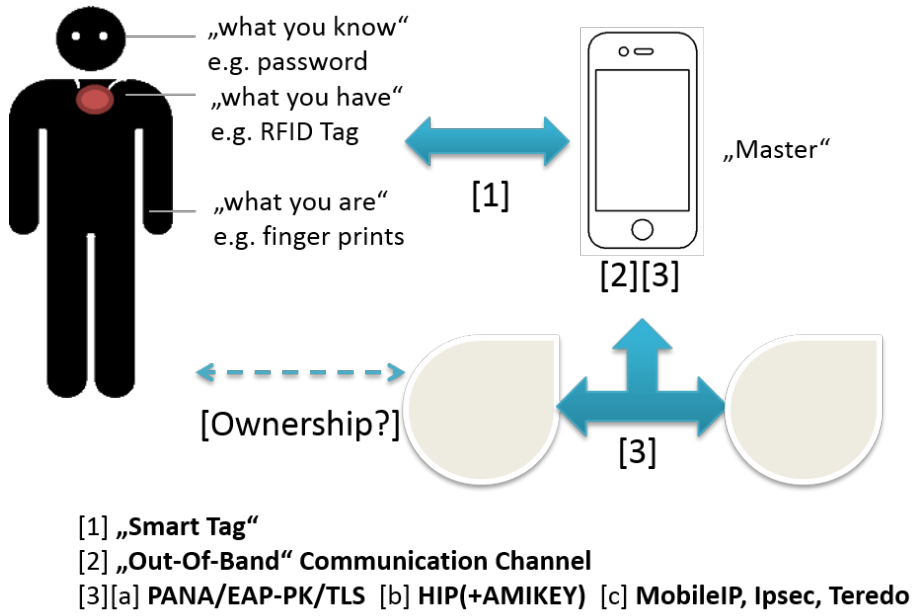

Figure 2.13: Ownership scenario - setup

To reach the WWW, they are dependant of a computationally more capable member of the intranet, which in this example is the user's smart phone. The smart phone will be therefore called "master" - resembling the master-slave terminology in Bluetooth communication[24]. The user, the smart phone and the two objects can communicate over different protocols and interfaces as follows:

- Person to Smart Phone. This is realised through the user interface of the smart phone. The ownership relation is more complex. The smart phone can be regarded as the virtual representation of the user[25]. The smart phone has to make sure that the owner is the one in physical possession. It is therefore favourable to authenticate the user with two factors that can be related to him. This is done by exploring the different authentication mechanisms (with a combination of "what you are", "what you have" and "what you know" attributes of the user). For example, RFID tags can be combined with PIN based authentication. The user wears an RFID tag somewhere on his body[26]. Per NFC, the smart phone can check on its

[24]The interested reader is referred to the Bluetooth specification in [B+01].

[25]This assumption has been made based on an interview made with telecommunication experts, see annex A. The reason is that a user's smart phone contains sufficient information to represent almost any attribute of the user, therefore it is adequate to hold the user's virtual entity.

[26]Passive UHF RFID tags can reach a 10 meter radius with a low energy consumption profile for the reader, if the amount of tags is low (this is the case for personal area networks), see [Dob12]. This makes the assumption reasonable.

booting and periodically thereafter, if the person can authenticate himself as its owner. The user has to register himself and his tag in the first boot process following the phone's "off-the-shelf" delivery or after a factory reset. The interested reader is referred to [FDW04] for more information on secure RFID authentication.

- Smart Phone to Objects. The smart phone has a special relation to the two objects as it is the intranet's master. At the same time, it is an object itself. Thus two connections are defined. Tag [2] in Figure 2.13 is an out of bound, i.e., a special communication protocol between the master and the intranet's slaves. This could be Bluetooth, for example. Before other objects are allowed to interact with each other inside the intranet, the out-of-band channel has to be established and master/slave relations have to be defined. Tag [3] is the communication protocol that is used between every object in the intranet. This could be, for example, a communication based on the Protocol for Carrying Authentication for Network Access (PANA), the Host-Identity Protocol (further discussed in Section 2.6.4) and a combination of other currently known network and authentication standards (e.g., mobileIP, Ipsec and Teredo).
- The communication of the rest of "things" with each other is based on the communication type of [3] as seen above.

Technologies. Figure 2.14[27] shows a further refinement of the scenario. Here, a special technology is chosen for every communication relation. The authentication relation from smart phone to owner, marked with [1], is realised with a RFID-tag recognized by the Near-Field-Communication capability of the smart phone and a PIN. The master-slave communication in [2] is realized by Bluetooth. This communication is specifically for master-to-slave messages, which can contain different commands, configuration or status requests. The communication of object to object is exemplified in [3] by the Host-Identity Protocol (HIP)[28]. This communication is used for exchanging operational data, e.g., for application related

[27]Figure 2.14 displays the Sony Smart Tag; Source: www.sony.com.

[28]Note: Host-Identity Protocol (HIP) protocol is still actively researched for the IoT domain, see Section 2.6.4 for details.

messages. Note that [2] and [3] are chosen to be out-of-bound in order to have dedicated communication channels.

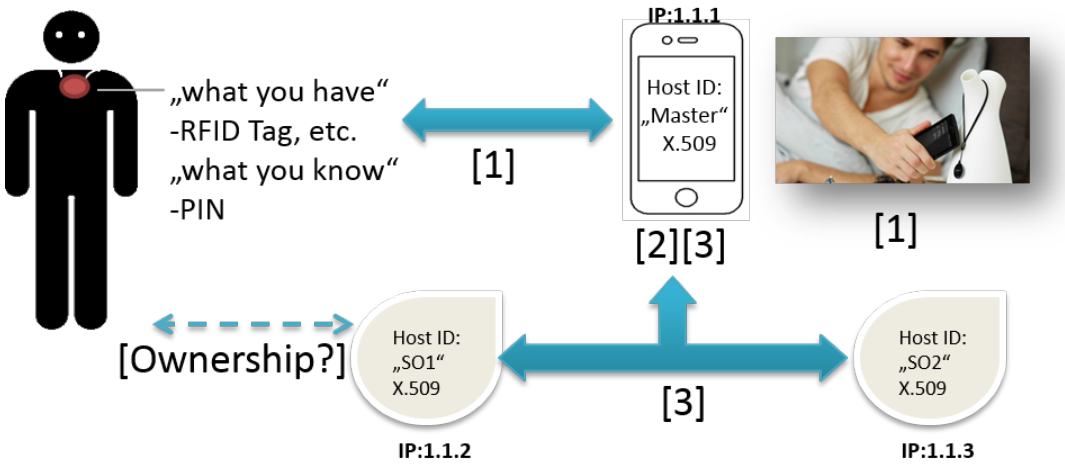

Figure 2.14: Ownership scenario with enabling technologies

The HIP can be used to establish an identity based communication inside the intranet. That means, the objects do not address each other by IP-addresses, but by identities. In a simplified form, identities are signatures over a random value, they can be verified and can be accounted to their master, where the master might serve as a certificate authority. The routing of messages is done via IP-address, the mapping of identities to IP-addressed is carried by the protocol. The Figure 2.14 displays an exemplary identification of the objects by the identities $SO1$ and $SO2$, which are verifiable via certificates of the form X.509. IP-addresses are obtained according to the master's subnet, with the master being the standard gateway to the intranet.

Protocol. Figure 2.15 further showcases the protocol. One of the two smart objects changes its physical possession from one person to another. In the previous Section it was mentioned that it is not expedient to verify if a human user is physically touching an object, but that it is enough to know if an object leaves the domain of authority of one user and reaches the domain of another. This domains are coined here as *Digital Shadows*. The term digital shadow has been used in the context of identity management before by Sarma et al., see [SG09]. In Sarma

et al.'s definition, the digital shadow is the area of influence of a user's virtual identity on one or more (web- or application-)services. This definition is extended here to the context of IoT, where the identity of the user influences one or more IoT objects. The mechanisms of influence are similar to those described by Sarma et al. (e.g., access control rules) with the addition of ownership mechanisms.

In Figure 2.15, object $SO2$ has obtained his identity from $master1$. $Master1$ is the virtual representation of $person1$, which the object now can be referred to. As described above, the identity is a verifiable cryptographic value and it is not computationally feasible to spoof or manipulate it.

In the example $SO2$ leaves $person1$'s digital shadow and looses its connection to $person1$'s intranet (specifically to $master1$). As soon as it enters $person2$'s digital shadow, it's identity is verified by $master2$.

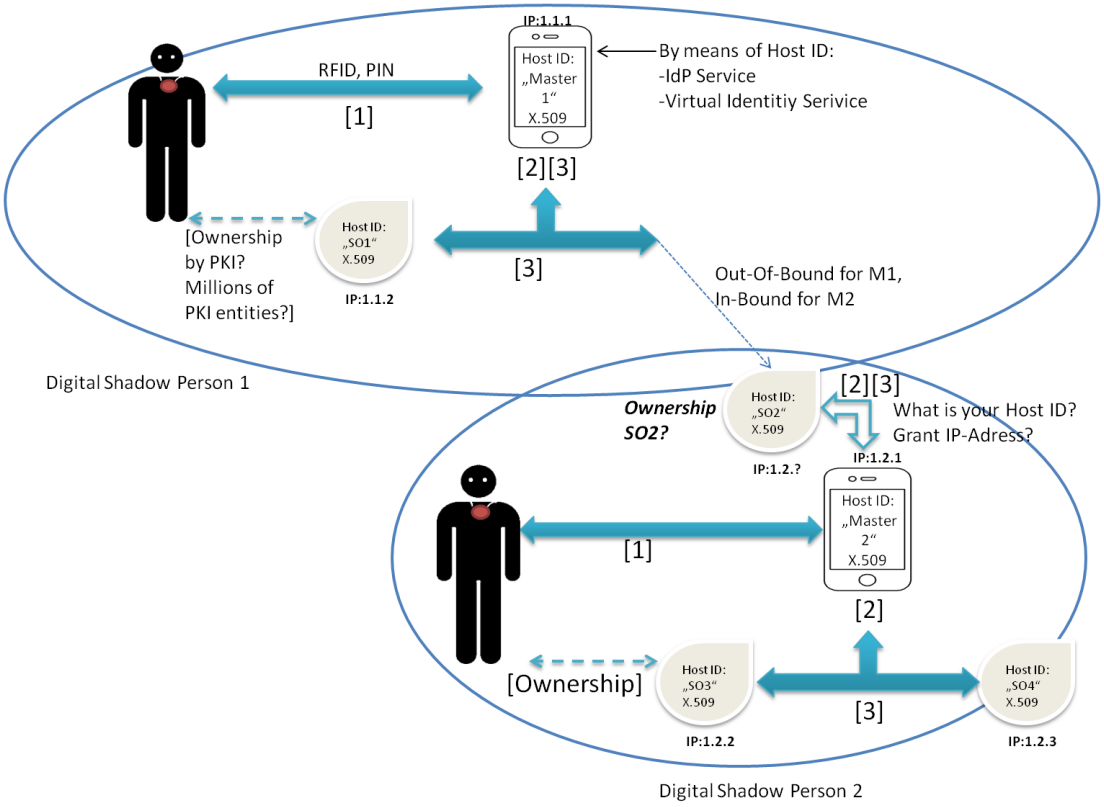

Figure 2.15: Ownership and digital shadow

During the identification process, $master1$ is identified as the previous *owner*, as the identity of $SO2$ if verifiable through public information of $master1$[29]. Note

[29] A possible mechanism that can be used here for the lookup is a public key infrastructure.

that *master*2 and *SO*2 use the same protocols [2] and [3] that *SO*2 used to communicate with *master*1.

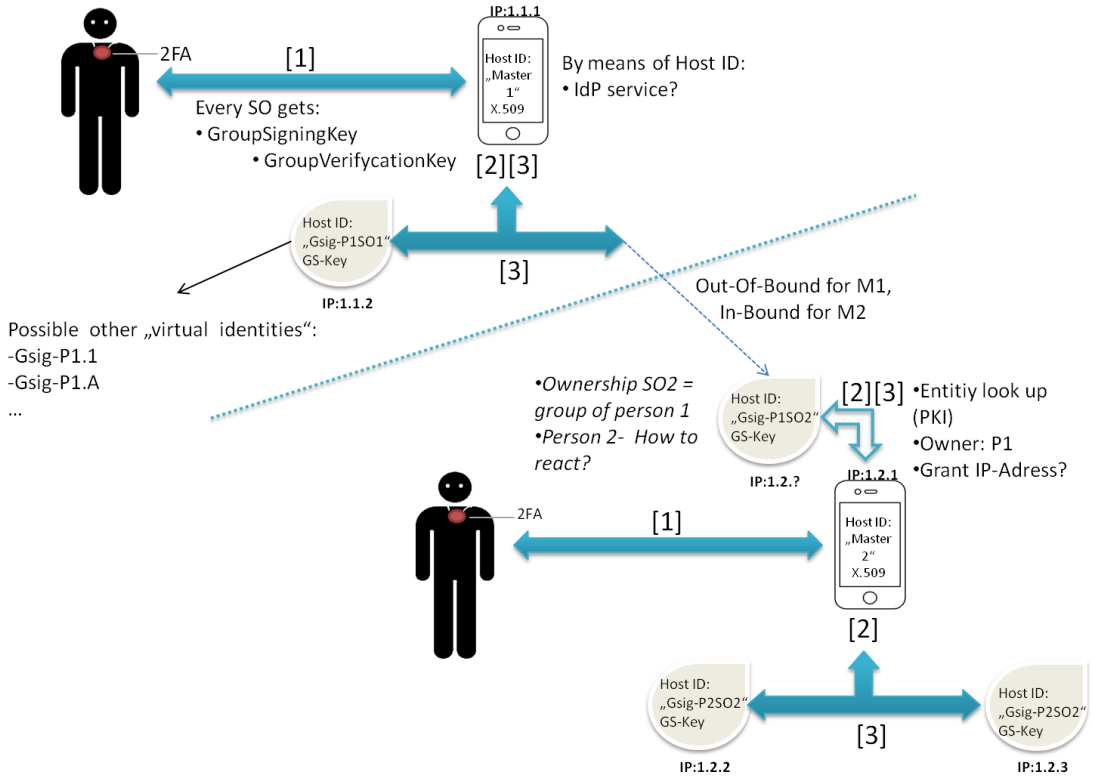

Figure 2.16: Ownership protocol example

Figure 2.16 shows a more refined version of the protocol. The identities are based on Group Signatures, where an object proves his identity by authenticating himself with a signature. The signature is publicly verifiable and proves that the signer is member of a specific group, see [MFG+12] for details. In the case of *SO*2, its signature would identify it as a member of the group owned my *master*1. The signatures are unlinkable, *SO*2 could identify itself as owned by *master*1 with different signatures against *master*2 and other masters, and at the same time staying unlinkable. A specific use case for the use of group signatures will be given in Section 4.3.4.

The protocol starts with the management of keys and certificates. Every object gets a member signing key and a group verification key, to be able to unequivocally identify other members of the group. The keys and certificates are managed by the *master*. Objects identify themselves by signatures as owned members of a group, which allows them to have several unlinkable identities due to the nature of group signatures. When an object changes its possession and enters a new digital

shadow, it authenticates itself with a signature. The master of the new digital shadow verifies the signature with the object's certificate either provided by the object itself or provided over a public resource identifier and a respective lookup for a trusted certificate authority. In the process, the new master can identify the old master and the respective ownership relationship. The new master (*master*2 in Figure 2.16) has to decide whether the new object may enter its digital shadow and participate in communication. The decision might be based on policies that where defined by the owner or through direct feedback, e.g., a push up notice by the smartphone. If the decision is positive, the new object may enter the new intranet and agree on how to communicate within the intranet and agree on possession based policies which the object has to follow.

Policies are an integral part of IoT and interplay by nature with ownership and possession. In the next Section, ownership, possession and policies are set into relation within the Rerum IoT domain model.

2.5.3.5 Ownership in the Rerum Domain Model

Section 2.5.3.3 introduced ownership and indicated how important it is for social behaviour and context awareness of objects in IoT. In order to resemble ownership in IoT, respective building blocks have to be defined and integrated in the IoT domain model. Figure 2.17 depicts the Rerum IoT domain model extended by ownership relations.

Ownership can be of two forms. Absolute *ownership*, where a user has absolute rights (and duties) over an entity, or, *physical possession*, where the user has only physical access to the entity, but might be limited in his actions by the rules of the (absolute) owner.

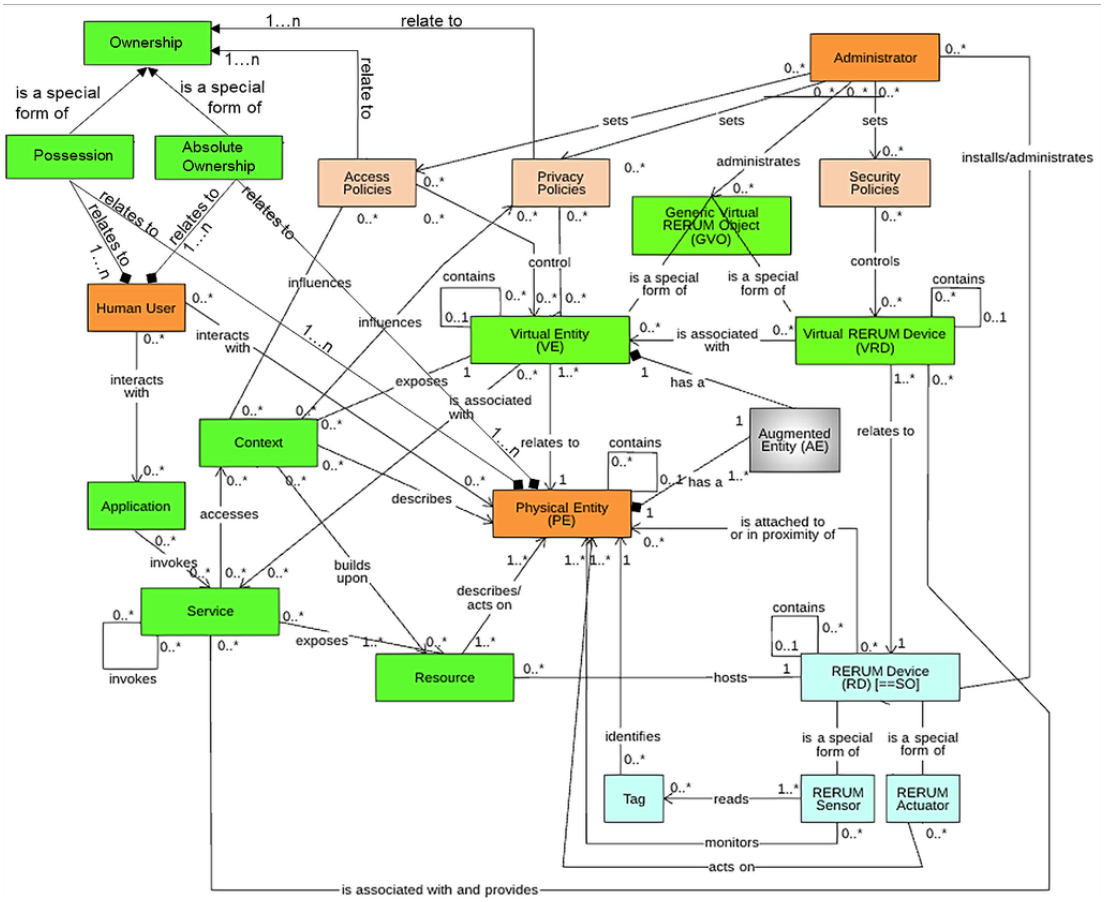

Figure 2.17: Integration of ownership in the Rerum domain model

Ownership is a relation between a human user and an entity. The relationship is therefore a composition between one or several physical entities and one or many human users, one for absolute ownership (a user owns/has the rights for one thing) and many for multi-party ownership (several users own/have the rights for one thing). Absolute ownership and physical possession may coexist as two compositions for the same entity and user. Ownership influences how users interact with entities in real-life, in IoT this is done via policies. Rerum defines two types of policies[30]: access policies and privacy policies. Although the domain model defines that an administrator sets (or defines) each policy type, it is the owner that the policies relate to. In both cases the administrator is either the owner himself or the administrator acts in the interest of the owner.

Ownership controls the access to the virtual entity by means of the policies, therefore no direct relation between VE and ownership is needed.

In the proposal for the establishment of policies, the HIP was mentioned as a

[30]See Section 2.4.3.1 for details.

possibly relevant technology. In the following Section, several technologies will be discussed that can be possibly used to implement the building blocks that were presented over the course of Section 2.

2.5.4 Implementation of the Rerum Domain Model

The Rerum middleware is an implementation of the Rerum domain model, as shown in Fig. 2.7, which follows the intranet / gateway based communication scheme, see Chapter 2.5.2. The middleware is based on the implementation of the OpenIoT platform which also follows the IoT-A ARM, see [SKH+15]. Rerum considers the following building blocks as the key to the implementation of the domain model (taken form [MTF+]):

1. Rerum Devices (RDs) that can be constrained or unconstrained devices, see the classification of Section 2.6.2. They equip one (or several) sensors and actuators. Rerum defines specific functional components for that run on the Rerum devices, called Rerum embedded mechanisms. Some of these mechanisms are used for privacy enhancement and are presented in Chapter 4.
2. Rerum Gateways (RGs) as part of the intranet / gateway based communication scheme. The gateways have some middleware functionality such as network and protocol translation (particularly from proprietary protocols on the devices to 6LoWPAN, as described in the IoT protocol stack, see Section 2.6.3) to communicate with the rest of the Rerum environment.
3. Rerum Middleware (RMW) that performs virtualisation (as formulated by the IoT-A ARM, see Section 2.6), performs transmission of data between devices and virtual entities and that provides publish/subscribe services for third parties.
4. Rerum Security Server hosts all security and privacy functional components. The security server can be a standalone component or an integrated part of the RMW.
5. Application Server hosts the applications of Rerum or is an external container for third party service providers.

The middleware consists of a collection of functional components in order to manage

virtual entities, data processing and the registration of Rerum and federated devices. These components are categorised in functional groups [MTF+]:

1. Service Manager handles requests from applications. For example, an application might need the temperature of a room and requests that Feature Of Interest (FOI) from the service manager. The Service Manager will resolve the FOI and map the virtual entities (and possibly virtual federations of entities) that relate to devices that are in physical proximity of the room. The process is transparent for the application. Additionally, it hides the devices, IP addresses and MAC addresses of the devices, such that the application provider cannot know exactly where the data comes from.
2. General Virtual Object (GVO) Manager handles the registration of Rerum devices and the creation of virtual entities by using predefined templates. The properties of the devices (sensors, actuators, RFIDs, context) are used to find the best match of a given template. The virtual entity is then registered in the GVO Registry.
3. Federation Manager is responsible for creation, composition and orchestration of federation. The federation manager could also support other forms of interaction[31] that require a leading entity.
4. Data & Context Manager processes the data received from devices. One of its components, the stream processor can either simply pass the data to VRDs and service agents or pre-process the data, e.g. perform map-reduce on a large data stream [YDHP07], data aggregation, the computing of mean, minimum or maximum values and other window functions. The data translator converts the stack of a device (e.g. 6lowPAN) to another stack needed by the application (IPv6 + REST). The context manager is a dedicated component that analyses data and, according to some rules [RER15], extracts the context surrounding that data.
5. Security Server which offers different components and technologies for security, privacy and trust that are deployed on the Rerum middleware and on the devices (SPT components). The security server holds them on the middleware. A full list can be found in [RER15, MTF+]. A partial list with

[31] See Section 2.5.1.

a more technical insight has been published in [TPS+15]. The components for privacy in all publications overlap largely with the technologies presented in Chapter 4 as they have been developed for Rerum and this thesis.

Figure 2.18 interprets the interleave of building blocks and functional groups.

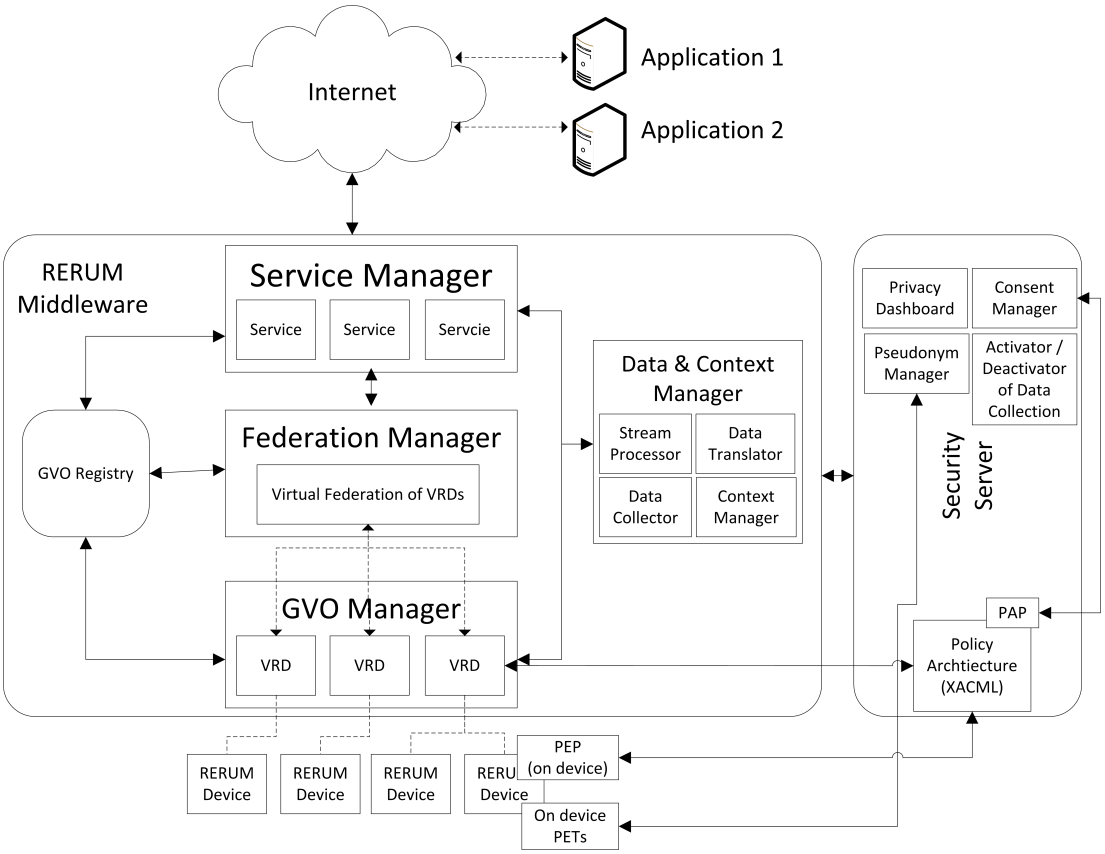

Figure 2.18: Rerum middleware

2.6 Technologies for IoT

In the previous Sections, conceptual and architectural building blocks of IoT were presented. Many of the building blocks are located in the devices, which are foreseen to be low cost and constrained in storage, computational as well as battery power. The reason for these constraints is that IoT devices have to be economic and very small as they will have to fit in everyday objects like home appliances (TVs, washing machines, fridges, ovens, ...), body wear (glasses, jackets, shoes, ...) and static objects like walls, windows and doors.

The size and economic constraints allow a significantly different "best effort" quality of services compared to portables, tablets and smartphones. IoT devices

change their location regularly, they have large sleep (passive) and wake-up (active) states. Networks formed by these devices have high information loss, low throughput and frequent topology changes. Therefore, technologies which have been developed for similar building blocks in ICT systems cannot be transferred to IoT directly. In this Section an understanding of why constraints exist and an overview of technology proposals for IoT will be given. This Section serves mainly as a snapshot of the state-of-the-art, as technologies are subject to change in the future. It should be noted that neither security nor privacy nor trust technologies will be reviewed in this Section, as these are part of the extended IoT domain model. Details can be found in Section 4.

2.6.1 Economical Value and Size Constraints

In order to exemplify why constraints on many IoT devices exist, current[32] developments in body wear and health care can be consulted for orientation. Health care providers have yielded a number of body tracking systems, such as smart bands to monitor the heart rate of users and smart shoes that can help to train balance and assess fall risks for elderly people, see e.g. [McC09] and [NAD+08].

The example of smart shoes can be used to analyse the costs and size constraints of an IoT device. In [Lor97], Lorand has broken down the costs of $70 shoes (non-smart, customer retail price)[33]. The shoes manufacturing costs are $20, thereof $9 only for materials. Operational costs and profit are amounted with $15.50, with $0.25 for research and development (R&D). The rest of $34.50 are the retailer's costs and profits. Following results can be obtained if the costs are set into relation:

1. Total costs (retail price): $70 (100%)
2. Manufacturing costs: $20 (29%) with $9 material costs (13%)
3. Operational costs and profits: $15,50 (22%) with $0,25 (< 1%) for R&D.
4. Retailer related costs and profits: $34,50 (49%)

[32] Products and prices were last checked on August 17, 2017.

[33] The breakdown by Lorand is from 1997, but comes quite close to similar breakdowns reported in 2014, see for example [Ben14]. In addition, Lorand adds more detail for every cost item.

It can be assumed that smart shoes retain a similar distribution of costs, although retailer (online vs. in-store retailers) and operational costs (supplier costs and profits) may vary.

To set IoT device costs into relation to the costs of the shoes, the following use case can be analysed: the product in [Ama14] is an additional module for a special brand of sport shoes. The module can be attached to the sockliner of the shoes and can be connected to any smart phone, tracking the pace, movement speed, distance, time and other parameters. The price for the product was calculated with $19 (recommended retail price). To break down the price structure of this product, a price calculation of electronic goods is needed. The price calculation of Kraemer et al. in [KLD11] will be used here. The calculation of the shoe sensor according to Kraemer et al. gives the following items:

1. Total costs (retail price): $19 (100%)
2. Manufacturing costs: $9,50 (50%) with $5,89 material costs (31%)
3. Operational costs and profits: $6,65 (35%) with $0,95 for R & D (5%)
4. Retailer related costs and profits: $2,85 (15% - electronic devices have a smaller margin)

The IoT device itself costs $5,89 for materials including assembling and $0,95 for R&D including software development and quality testing, making a total of $6,84. Compared to the overall price of a pair of smart shoes (shoes price $70 plus sensor kit price $19) the device makes up 8% of the total product costs including R&D[34].

Table 2.1 summarizes the cost breakdown and gives insight to the size related constraints. In addition, a device from the popular Raspberry family ($40 retail price, material and R&D costs estimated with 36% according to [KLD11]) is added as a comparison to the Table. The Raspberry device is far more powerful than the IoT Device and would support the application of many existing privacy and security technologies. At the same time the Raspberry is several times larger in size and costs, making it arguably difficult to use in the Smart Shoes and many other IoT scenarios.

[34]In cost and performance accounting, the operational and retail related costs are assigned directly to the individual products, but in this case, they are excluded: these costs can be combined when shoes and device are sold together and reduced in the final product. Manufacturing and R&D costs are directly related to the shoes and devices respectively and cannot be combined, giving thus a much better indicator for proportions of costs.

	Smart Shoes (size 42)	IoT Device	Raspberry
Total Costs	$89 (100%)	$6,84 (8%)	ca. $14,4 (16%)
Product size	ca. 27 x 15 x 10 cm (100 %)	2,4 x 3,5 x 0,8 cm (> 1%)	12,7 x 10,2 x 7,6 cm (24%)

Table 2.1: Comparison of size in relation to costs

The price and size data was taken from [Ama14, Ama16b, Ama16c].

2.6.2 Classification of IoT Devices and Networks

The IETF has classified constrained devices by their computation and storage capacities in [BEK14], Table 2.2 shows the individual classes.

The most used IoT devices have been identified in "class 1", with a working memory restriction of *10 KB RAM* and a storage capacity of *100 KB Flash*, see [Bor15]. Class 1 devices are typically powered by coin or dry cell batteries with a maximum capacity of *2376 joules* or *0,66 watt hours*[35], see [Dev11]. Networks that consist mainly of devices of class 1 or less are called low-power and loss networks. The IETF define this networks as *"typically composed of many embedded devices with limited power, memory and processing resources interconnected by a variety of links [...]."* The classes are not clearly cut and other classes are used in some cases for IoT. For the elicitation of adequate privacy enhancing technologies, the constraints of class 1 are those that will impose a significant factor. We will look into details in Chapter 4.

[35]The lifetime of the battery depends on the device actions and on the sleep and wake up times of the device. This makes it difficult to estimate a reasonable average lifetime.

Name	Data size (e.g., RAM)	Code size (e.g., Flash)
Class 0	less than 10 KB	less than 100 KB
Class 1	10 KB	100 KB
Class 2	50 KB	250 KB

Table 2.2: IETF classification of constrained devices

2.6.3 IoT Protocol Layer Stack - IETF LLN

The IETF has developed and proposed several technologies as part of the Routing Over Low power and Lossy networks (ROLL) working group to build an IoT protocol stack. The proposal retains the layers of the TCP/IP protocol stack, Figure 2.19 shows the comparison.

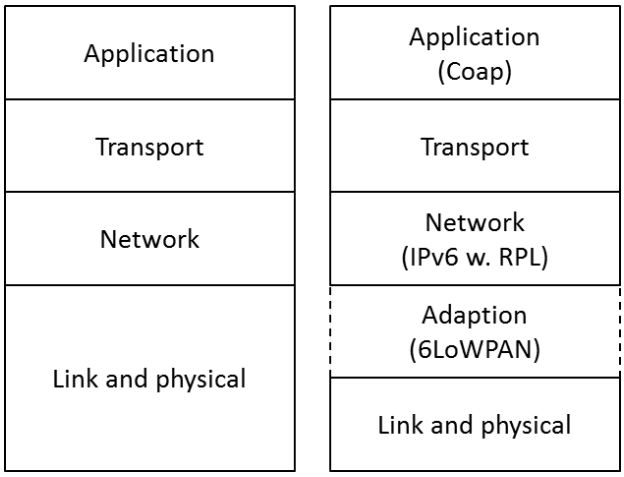

Figure 2.19: Comparison of the TCP/IP stack (left) and the IETF IoT LLN stack (right)

The upper layers are represented by a respective working group in the IETF, where as the link and physical layers are addressed by the Institute of Electrical and Electronics Engineers (IEEE). The Low Power and Lossy Networks (LLN) stack differs primarily from the TCP/IP stack due to the addition of the adaption layer. The layer was introduced due to following constraints:
For low-power devices, such as wireless sensor networks (WSNs) and wireless personal area networks (WPANs), the protocol family IEEE 802.15.X has been widely accepted as the standard for low-power radio transmission, see [BPC+07, LKR04]. The IEEE 802.15.4 protocol was designed for body and personal area networks, two networks that are very prominent in IoT. Therefore, the IEEE 802.15.4 protocol is mentioned specifically in the stack. The IEEE 802.15.4 protocol defines the architecture for the link and physical layers with a maximum payload size of 127 Bytes, see [Ass11].

The upper layers of the TCP/IP stack, specifically based on the IPv6 standard, define a header overhead of 68 Bytes for the TCP and IPv6, UDP, ICMP and

TCP headers, which takes over fifty percent of the possible payload transmission. Additionally, the maximum transmission unit of the IPv6 standard requires 1280 Bytes which differs strongly form the 128 Bytes maximum frame size of the IEEE 802.15.4 protocol. This discrepancies are resolved in the adaption layer. As mentioned before, the IEEE 802.15.4 standard is envisioned as the protocol for the link and physical layers. The adaption layer is covered by the 6LoWPAN protocol (the acronym stands for "IPv6 over Low power Wireless Personal Area Network"). 6LoWPAN resolves the overhead by encapsulating the IPv6 headers, fragmenting them and occasionally compressing the headers if possible. A full descirption can be found in [MKHC07]. An overview of the whole LLN family can be found in [ICT+13].

2.6.4 IoT Protocol Layer Stack - IETF LLN + HIP

In Section 2.5.3 the host-identity protocol was mentioned. The host-identity protocol (HIP) was firstly described by Moskowitz et al. in [MNJH08] in 2008 and has been proposed as a key technology for the IoT stack in [Uri09] in 2009. The idea has been recently picked up as a possible addition for the IoT LLN stack in [VMZS+13, GMKK+13]. Also, a new version of the protocol is being proposed in [MHJH15a].

The main idea of the HIP protocol is to separate host and network identifiers. Host identifiers are used to identify network participants, as seen in the domain name system. Network identifiers describe the address of the participant in order to identify where network packages have to be routed to, as with IP and MAC addresses.

In traditional networks, the IP address serves as both, the host and the address of the host. If a participant changes his IP address, he cannot be identified and reached in the network unless an agent updates his address (as seen in the mobile IP protocol [Per98]) or the participant itself propagates its new IP address. Occasionally he can use a higher level protocol for authenticating himself as the previous participant.

HIP adds an additional host identifier layer to the TCP/IP protocol stack to allow the separation of the two identifiers. An application would now identify

a network participant by his host-ID, which remains in the network even if the participant changes his location an IP-address. The host identifier is an authenticity credential, it has to be generated by cryptographic material to avoid masquerading as an arbitrary participant. The HIP has defined the *HIP key exchange* for this reason, one of the main mechanisms of the protocol. Figures 2.20 and 2.21 show a simplified comparison of IP-based and HIP-based communication.

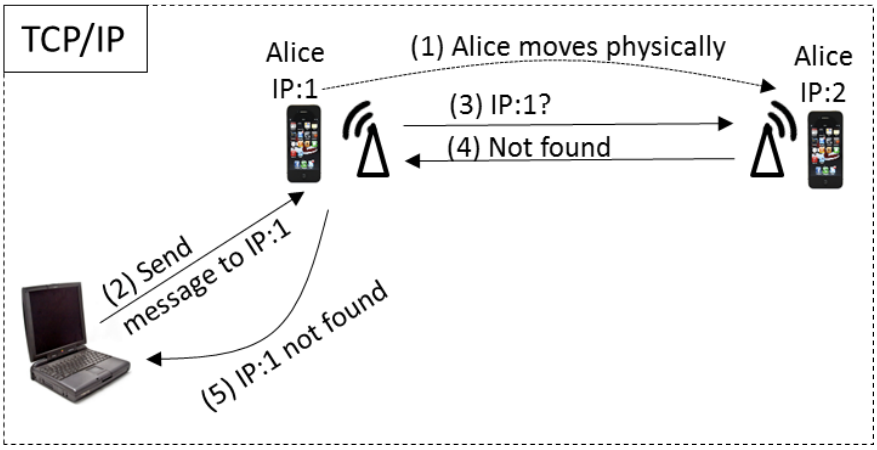

Figure 2.20: IP based communication

Figure 2.20 depicts a scenario where two parties, Alice and a Server, communicate over an IP-based network. The communicating parties use IP-addresses to send and receive messages. In case of an IP change, e.g. due to physical movement of Alice (step 1), the Server will still send messages to Alice's old IP-address (step 2), as this is the only identification the Server has from Alice. Consequently, the network will not be able to deliver the message, as Alice's IP-address is now unused (steps 3 and 4). The Server will receive an IP not found notice from the network (step 5) and close remaining resources, e.g. sessions, instances, etc.

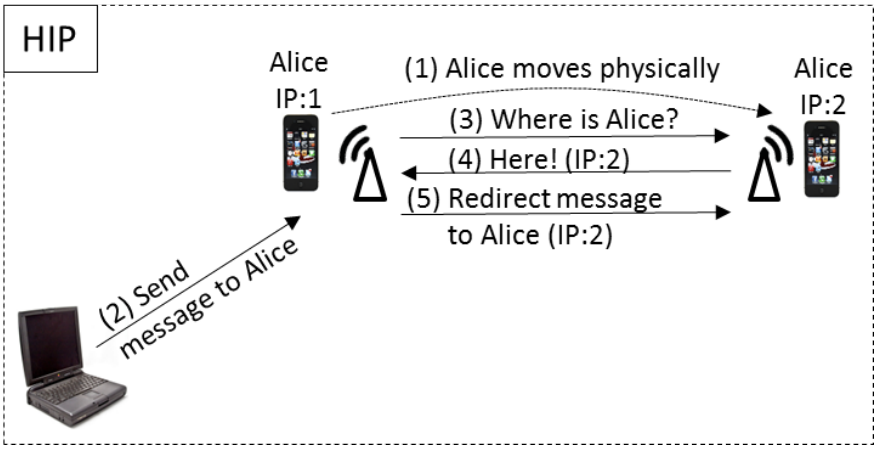

Figure 2.21: HIP based (bottom)

Figure 2.21 depicts the same scenario where Alice and a Server communicate over an HIP-based network. HIP participants exchange unique and verifiable identities via the HIP key exchange.

The communicating parties use their identities to send and receive messages. The network itself is able to validate the participant's identities as well, therefore able to reliably route messages (over IP-addresses). In case of physical movement of Alice (step 1), the Server will still send messages to Alice's identity (step 2), as this is the only identification the Server has from Alice. Consequently, the network will try to find Alice by her identity, which is possible as soon as Alice reconnects to the network. The network will be able to deliver the message, as Alice can be identified and her IP-address can be resolved (steps 3, 4 and 5).
Although several protocols integrate host related concepts to allow updating a participants position, the HIP offers a set of attributes that can be helpful for IoT settings:

- The HIP allows end-to-end encryption and integrity protection, as the HIP key exchange can be used to derive further keys for respective protection. This is similar to the use of Datagram Transport Layer Security (DTLS), as HIP end-to-end protection does not rely on state management.
- In case of message overflows, participants can quickly change their identities without changing the transport or physical layer attributes. Compared to other denial of service mitigation mechanisms, this is an effective and

simple mechanism against denial of service attacks that supports resource constrained IoT devices.

- The host identifier is a value with high entropy by design. Host identifiers could be used as a pseudonym for network participants to enhance their privacy, even using multiple pseudonyms for the communication with multiple participants. The end-to-end protection remains also with multiple pseudonyms.
- Host identifiers can be used in combination with group authentication attributes. That means, a network participant (e.g. an IoT device) can authenticate itself as a member of a certain group. This again can be used to resemble real-life ownership / possession relationships[36].

The first version has been discontinued and has been given an errata by the Internet Engineering Steering Group (IESG) for using obsolete or inefficient cryptographic primitives which could not be changed or adapted. The full IESG errata was publicly added in the first HIP draft by Moskowitz et al. A second version was published in April 2015, see [MHJH15b], which addresses the concerns of the IESG, adds crypto agility[37] and implementation related changes. In comparison, although many of the properties of HIP are useful for IoT, HIP has not been thoroughly researched as the protocols and technologies found in the LLN stack and it remains to prove that HIP can be an efficient and reliable alternative. As of August 17, 2017, the an errata exists for the second version of the HIP in [MHJH15b] as well.

[36]For example, a device is owned by a subject versus a device is in physical possession of a subject. Details on ownership and possession are can be found in Section 2.5.3.

[37]Crypto agility is the ability to easily switch from one cryptographic primitive to another.

Chapter 3

Privacy in the Internet of Things

The fundamental conflict between IoT and privacy was pointed out in the introduction. The introduction also gave an insight to possible consequences of data breaches and the challenges of applying data protection to future IoT systems.

The conflict stems from the significance of data. Data is an asset of immense business value for companies, but so is the trust that customers and users have in them. Privacy violations may endanger both, the value of the data and the trust that people have placed. Moreover, the costs of privacy related lawsuits could be huge as will be discussed later. Yet, privacy breaches and violations are not uncommon, although they are not always put in public focus. For instance in 2009, Facebook settled a privacy related lawsuit over $9.5 million due to its service Beacon. Beacon tracked a user's behaviour at 44 external shopping websites and published the consumption activities of the user in its Facebook profile under the "News Feed", see [Per08]. Beacon was introduced in Facebook as an opt-out approach and needed no explicit user consent. Following the lawsuit, Facebook tried to implement further privacy controls and easier opt-out, see [Zuc07], nevertheless, Beacon was shut down shortly after that.

Anthem Inc., second largest insurance company in the United States, leaked 80 million records including names, social security numbers, dates of birth and other sensitive details such as health status [MY15].

Google Buzz is a further example. Buzz was a social online network that was supposed to compete with Facebook and Twitter [Gro10]. Google took advantage of its Gmail user base and linked both services to generate a high user traffic.

Based on the email contacts of the users, Buzz profiles showed user names and relationships between each other.

The latter was the most intrusive property of Buzz, as email relationships of users were exposed publicly. Google was confronted with high criticism, including a lawsuit filed by The Federal Trade Commission, which was settled for $8.5 million, see [Act10].

In Chapter 1 the costs per breach and the outcome for users and companies was outlined. Companies might not see the need for data protection, as they can calculate the costs of a breach and use administrative strategies to mitigate the consequences.

A possible way to solve this indifference is to adapt regulations to demand proper assessment of adequate privacy protection in companies and organisations and calculate fines accordingly. That means that regulative directives have to demand for protection and at the same time give guidance on how to do so. Complimentary privacy guidelines through privacy principles have been proposed by research communities in the past, which are found in a similar form in current privacy protection regulations. New proposals may serve as the basis for a modern privacy regulation as well. In order to assess the current picture, current data protection directives, their possible problems and additional scientific proposals will be discussed in Section 3.2.1.1.

One of the main problems in data protection is the interpretation and the integration of those guidelines and principles in the lifecycle of a system. In the following Sections, engineering frameworks which support that integration are discussed and followed up with a possible PDLC in Section 3.3. The current state of regulations and the possibilities of privacy engineering will be mapped to the picture of IoT. The mapping will use the characteristics that were outlined in Section 2.2. The next Section will examine the motivation and role of different stakeholders that participate in data protection for IoT.

3.1 Stakeholders

Elias et al. underline in [ECJ02] that every R&D project affects different stakeholders in a different way and that a proper analysis can be crucial to the success of a project. Privacy engineering and privacy research for IoT are branching projects in many directions; the following (incomplete) list of stakeholders should set the first stone towards a stakeholder analysis by touching on the topics: dynamics of stakeholders, rational (who is interested and what are their motivations), process (which relationships exist) and transactional (how to approach) levels, as proposed in [ECJ02].

Chapter 2 introduced *users* and *companies* as two of the main stakeholders.

A third type of stakeholder are (privacy) *researchers*. Researchers may be users themselves and are generally interested to exploit the possibilities that arise with new technologies. Within that process, they contribute to an essential part in making transparent what is viable for privacy protection in form of concepts, architectures, technologies and other wise.

A fourth type of stakeholders stems from *jurisdiction*. Jurisdiction is a main pillar of data protection and is used as the core in business environments.

Throughout the course of this thesis, all of this stakeholders have given input on their point of view of privacy protection. The following list summarizes the input.

Companies. The viewpoint of companies range broadly based on their interests. For brevity, the two most heterogeneous are considered: companies that see data protection as a business differentiator and companies that see data protection as legal topic that is mostly handled in agreements and contracts. Companies that aim to differentiate themselves with privacy topics have a very profound knowledge of privacy enhancing technologies and know how their service could affect the privacy of users. Services provided by these companies are offered with different paying models or rely on donations in favour of renouncing to the extensive collection of user data. These companies often replicate other popular services that are data intensive, e.g. on-line social networks or search engines, but try to provide a similar

quality without being intrusive to the privacy of users. Privacy engineering is motivated on a voluntary basis through open collaboration.

Companies that handle privacy as a contractual issue rely heavily on consent[1]. Legal departments of these companies strive to formulate service agreements that use consent to cover all aspects of processing user data during service provision. Privacy research is given lesser priority or is rejected unless an incident involving user data has impacted the company. Involvement in privacy research and engineering may be motivated by the technological advancements in impact areas that they act in. Additionally, concepts like privacy by design and new regulation may further motivate data protection, see Section 3.2.1.3.

Users and Researchers. Users may have different experiences and awareness regarding data protection. Again, the two most heterogeneous types of users are considered: users that have little awareness of the impact of privacy and users that are highly aware and behave accordingly.

Users that have little awareness maybe indifferent towards privacy research. On the one hand, they may argue that companies have the last word on the intrusiveness of services or that the disclosed insight about their privacy is insignificant. These users have shown to change their viewpoint on privacy when shown the criticality of privacy breaches as mentioned in Section 1.1. The interested reader is referred to the elaboration of Solove in [Sol07] for further details on this topic.

Users with a high awareness on privacy impacts are careful with their behaviour in ICT systems and their choices regarding companies and services. These users, possibly privacy researchers themselves, have a high technical understanding and generally support privacy engineering and privacy research. These users deliberately choose service based on their policies and tend to avoid intrusive services. This behaviour positively impacts privacy as a business differentiator. On the contrary, the stringent orientation towards data minimization may cause an objection of services that rely on data exchange. For example, users of this category have objected privacy research

[1]The concept of consent is explained in the following Sections.

as a whole in the impact areas described in Section 2.1 as they see them unnecessary, due to the fact that they fundamentally object the services and the impact areas themselves (´´who needs a talking fridge?"). In this case, the socio-economical problems that are found in impact areas and the benefits of data exchange therein, as shown e.g. in healthcare, can help to motivate the participation of these users as consumers and researchers.

Legislative stakeholders. Legislation sets the framework and motivates companies to comply with data protection. Privacy research and legislation have to work together in order to establish organisational and technical best-practices that help to determine if protection is done adequately. More on the synergies of legislation and privacy research is elaborated in Section 3.3.9.

3.2 Privacy Engineering

Compared to privacy, security in software and system engineering is a well known domain today.

Similar frameworks to support privacy from the design of a system will be discussed in the following Sections and followed up with a possible privacy development lifecycle framework in Section 3.3.

3.2.1 Privacy Principles

There are several well-established security engineering principles, best practices, and guidelines for software and system development. Frameworks or methodologies like the Security Development Lifecycle, see [HL06], the Building Security In Maturity Model, see [MCM09] and the Software Assurance Maturity Model, see [CD12], can help developers to have a clear overview of their security from the design, identify the assets in their systems, assess threats and risks, as well as develop mitigation and benchmarking tools to test their security in a verifiable and comparable way.

In the case of privacy, the situation is not clear-cut. There are several so called "key approaches", "manifestos" or "foundational principles", and guidelines like *"Privacy Engineering"* by Ian Oliver [Oli14] and *"The Privacy Engineer's*

Manifesto" by Dennedy et al. [DFF14]). The approach by Denny et al. aims at companies and resembles (and also cites) security standards such as the ISO 2700X standards family. The approach does not include any threat and risk based measures, even those recommended in the standards, which are a best practice known in security lifecycle development. Oliver's work uses ontologies and a very high level description of what privacy maybe to reach a common terminology, but there is still no consensus on how a possible privacy development framework should look like, nor what the main engineering principles would be, nor how an engineer could implement those principles. The interested reader is referred to a case in point analysis on this issues by Rubinstein et al., see [RG13]. Further foundational principles for privacy can be found in the privacy by design framework, see [Cav09a], which will be briefly presented later. Considerable privacy engineering proposals can also be found in PRIPARE [Not15] and in the privacy guidelines in Spiekermann and Cranor [SC09]. These will also be presented and discussed in the remainder of this Chapter.

According to the Internet Privacy Engineering Network "*one reason for the lack of attention to privacy issues in development is the lack of appropriate tools and best practices [...]. There are, unfortunately, few building blocks for privacy friendly applications and services[...]*", see [BW15]. Some building blocks and privacy enhancing technologies for the Internet of Things that were developed during the course of this thesis are formulated in Section 3.3.3. The building blocks were designed as proof of concepts for privacy technologies in IoT and to support privacy in the four main use cases of Rerum.

In the remainder of this Section, the current approaches to Privacy Engineering, including Privacy by Design and PRIPARE, will be discussed. Also, it is sketched how a PDLC based on security best practices should look like.

First privacy principles from a regulation point of view will be looked at.

3.2.1.1 The European Data Protection Rules

The basis for European Data protection is the EU directive of 1995. It adopted many principles of fair information practices, which are basic guidelines for the processing of electronic data, see [Rei94]. Similar principles can be found also in

the Canadian Privacy by Design model. The principles have been updated in 2002 and can be found in two directives, Directive 95/46/EC, see [Dir95], and Directive 2002/58/EC, see [Par02]. The directive of 2002 is an extension of the privacy protection rules of 1995 for privacy in electronic communications. In summary, the principles are:

Consent states that personal data shall be collected and processed only for a specified, explicit and legitimate purpose. The words "specified" and "legitimate" imply that a data subject and the processing party have to agree on a common consent to how the data is exactly processed and which processes are outside of a legitimacy.

Purpose legitimacy and specification is closely related to consent. This concept demands service providers to understand and specify how specific personal data is used and for which purpose.

Collection limitation means that the collection of personal data must be "adequate, relevant and not excessive".

Data minimization is intertwined with the previous principle. Data minimization heavily supports privacy by design, by helping to avoid the collection, generation and storage of personal data.

Notice and Access is defined as *"communication to him [i.e., the data subject] in an intelligible form of the data undergoing processing and of any available information as to their source, [and the] knowledge of the logic involved in any automatic processing of data concerning him at least in the case of the automated decisions[...]"*.

Individual participation according to the European Directive 95/46/EC on Protection of Personal Data (art.12 (b)). Data subjects have the right to withdraw from the processing of their personal data, including data collection.

Accountability points at data subject's right to receive compensation from a processing party in case of data breaches. Therefore, data controllers have to be identified clearly and responsibilities have to be assigned for the data they are controlling.

The principles express fundamental rights of all EU citizens and have helped to harmonize data protection regulation in Europe as they are widely applicable due to their neutrality in terms of technology. On the other hand, some aspects remain unspecific. For example, it is unclear how much choice and control citizens should have.

In [Koo14], Koops discusses the directives from a citizen perspective and observes that exercising data subject rights is *"highly theoretical. Yes, you can be informed, if you know where to look [...]. Yes, you can request controllers [...] if you know that you have such a right in the first place [...]. Yes, you can request correction or erasure, if you know whom to ask (but how are you ever going to reach everyone in the chain[...] ?). There are simply too many ifs and buts to make data subject rights meaningful in practice."*

In 2012, a review of the Directive 95/46/EC and Directive 2002/58/EC had started to create a new European Data Protection Regulation. The new regulation should target the difficulty to apply the protection rules and the unclear scope of the previous regulations by defining more specific and additional regulations. The new regulation is also envisioned to replace directives and regulations of local European governments, such that following one regulation ensures data protection over the whole European Union.

3.2.1.2 European Data Protection Rules - Review 2012

The review of 2012, entitled *"Proposal for a Regulation of the European Parliament and of the Council on the Protection of Individuals with Regard to the Processing of Personal Data and on the Free movement of Such Data"* was reviewed and finally released in December 2015 as the General Data Protection Regulation (GDPR), see [C+12a]. The regulation governs every country of the European Union and consists of two instruments, the data protection directive and the general data protection regulation, see [Com15a].

The directive serves as the basis for law enforcement. The regulation adds several new principles and makes previous principles more accessible. The EU lists the following principles as the most notable[2]:

[2] Adapted from [Com15a].

Easier access for a subject's own data. Individuals have more information on how their data is processed, this information should be available in a clear and understandable way. The implementation of this principle in highly distributed systems such as those of IoT, where the linking between systems and databases generates privacy relevant information, remains technically and organisationally challenging.

The right to data portability. The EU denotes that it will be easier to transfer a subject's personal data between service providers. This happens only at a legal level, the technical implementation of the transparency demanded in the first rule, specially throughout several sub-processors, remains challenging.

Clarified "right to be forgotten". The directive states that when a subject no longer wants his data to be processed, and provided that there are no legitimate grounds for retaining it, the data will be deleted. The challenge, specially for IoT systems, remains in the verification of such a process. Service providers need to have a deep understanding of their systems, and perform assessments on where and in which circumstances personal information may be generated. The wish of a user to delete his personal data from the system may trigger a complex (and legally challenging) transparency process throughout several sub-systems and sub-contractors.

The right to know when a subject's data has been hacked. Companies and organisations must notify the national supervisory authority of serious data breaches as soon as possible so that users can take appropriate measures. The terminology used here (*"serious"*) leaves room for possibly conflicting interpretations.

Companies and business organizations are provided with new legal grounds to build privacy protection. Noted are[3]:

One continent, one law. The regulation establishes one single set of rules in the EU. The rules are stated to *"make it simpler and cheaper for companies to do business in the EU"*.

One-stop-shop. Businesses will report to one single supervisory authority. This is estimated to save 2.3 € billion per year.

[3] Adapted from [Com15a].

European rules on European soil. Companies based outside of Europe have to apply the same rules when offering services in the EU.

Risk-based approach. The risk based approach resembles the best practices as known in security life cycle development.

Rules fit for innovation. The regulation motivates *"Data protection by design"*. Privacy-friendly techniques such as pseudonomysation are encouraged and may be a measure to decide weather proper data protection is applied in a company or not.

While the new directive and regulation set a promising new legal ground, the implementation of the according processes and the support by adequate technologies remain a big challenge. Solove describes in [Sol15] further implications.

Penalties. Article 79 of the GDPR defines penalties for non-compliance to the regulation and the supervising authority, stating that the penalty may be either up to 10€ million or 2% of the total worldwide annual turnover of an organization or company for infringements regarding the regulation. Also, up to 20€ million or 4% of the total worldwide annual turnover are demanded for non-compliance adjudged by the supervising authority. Solove underlines that the infringements concern basic rules such as conditions for consent as well as transfers of personal data in non EU-countries. These new penalties may also serve as a motivation, as companies and organizations may regard the high fines as potential risks to their businesses.

Cross border data transfer. Soloves sees a blurry situation regarding cross border data transfer. The GDPR requires countries outside of the EU to have an adequate level of protection, but a solution to the judgement that brought down the previous agreement on cross border data transfer ("Safe Harbour agreement"), is still pending. In the judgement of the so called "Schrems vs. Facebook" legal case, the weak protections against access to personal data by the US government was the main complaint, see [CG15] for details. The GDPR and the Schrems case will have some kind of impact to a new Safe Harbour agreement, but technical means, interpretation and reference judgements will be needed to create a solid legal ground.

Vendor management. Article 26 of the GDPR states that a company or orga-

nization that handles the personal data for another organization or company, may not give the data to third party or sub-contractor without the consent of the organization or party where the data came originally from. Although this article sets a solid ground for transparency between organizations and their sub-contractors, the problem of consent management (which also affects users) still remains. More on the structural problems of consent in data protection will be discussed later in this Chapter.

Role of the Data Protection Officer. Article 35 and 37 describe when a privacy protection officer has to be designated and what the roles of the officer are. Article 36 states that the privacy protection officer has to be "properly and in a timely manner involved in all issues which relate to the protection of personal data." This ensures that the privacy protection officer is actively involved in the operational processes of the organization and companies.

Training. The tasks of the privacy officer described in Article 37 of the GDPR comprise "awareness raising and training of staff involved in the processing operations." Consequently, a process has to be established in organizations and companies to assess when staff is involved with data processing and when trainings have to take place. This requirement is elaborated further in the proposal on a privacy development life cycle in Section 3.3.

Consent. Consent remains one of the main pillars of privacy regulation. Consent establishes the relation between a data subject (the subject whom the data relates to) and the data controller (the party that collects and processes the data) and defines under which circumstances the data subject agrees on the processing of his personal information. In Article 7, the GDPR defines that the data controller carries the burden of managing consent management. The consent of each data subject needs to be made identifiable by demand and needs to specify why the personal data is needed for service provision. Again, this demands a process in a privacy lifecycle. Further details are given in Section 3.3.

The Right to Be Forgotten. The EU announcement in [Com15a] already declared this as one of the most notable rights in the GDPR which is defined in Article 17. It was already mentioned that the technical implementation

is the main challenge of this new right. Article 17 recognizes the problem and requires that data controllers: "tak[e] account of available technology and the cost of implementation, [they] shall take reasonable steps, including technical measures". Again, the GDPR sets a new requirement that demands an assessment of the technical possibilities.

Data Breach Notification. Two notifications are defined. Article 31 requires reporting of data breaches to the supervising authority. Article 32 requires notification of data subjects in case of breaches that are "likely to result in a high risk to the rights and freedoms of individuals." Solove underlines the requirement of a "high risk" and states that EU authorities need to define what a high risk means. Independently, companies and organizations will have to assess data protection breaches with a risk based methodology. This requirement makes it difficult to follow privacy frameworks that do not consider a risk based approach, such as [DFF14].

Scope. Article 3 states that the GDPR applies for organizations and companies that collect and process personal data in the European Union, even if the processing itself happens outside of the EU.

Solove sums up his observations by stating that the GSPR is a stricter regulation than the US privacy regulation. A stricter privacy regulation has the implication of possibly hurting data transfer and economical relationships between the US and the EU. Solove also mentions another perspective: the EU is leading the development of privacy regulation and protection. The result: international companies tailor their services and business to EU regulations and EU terminology more than any other regulation, including that of the United States.

Solove mentions that privacy protection may have benefits and may serve strategic goals. Cavoukian has also named several favourable implications for companies and organizations in her framework called privacy by design. Privacy by design or similar protection concepts are often cited in data protection, such as in Article 23 of the GDPR ("data protection by design and by default"). Cavoukian's proposal is discussed in the next Section.

For the Internet of Things, the cross border regulations and shifting the burden of correct consent management to the data processors are key points of the GDPR.

But the GDPR misses to renew the concept of consent which causes several structural and cognitive problems, see [Sol12]. Also, the mentioned multi-party consent, a core problem of the Internet of Things, is not given a clear legal interpretation.

3.2.1.3 Privacy by Design

Privacy by Design (PbD), see [Cav09b], has been suggested as the solution for data protection for several years now. The term "privacy by design" is claimed to be introduced in [HB95] in 1995[4], but the term itself does not appear in the cited publication. The first trackable combination of the terms "design" and "privacy" are found in the work of Bellotti et al. called *"Design for Privacy in Ubiquitous Computing Environments"* in 1993, see [BS93], which already discussed privacy implications of ubiquitous systems and the challenge of adapting the term privacy to evolving technologies.

The most advertised privacy by design framework today is the so called Canadian framework, which compiles a number of privacy principles.

These principles resemble other fair practice principles, such as those stated by the Organisation for Economic Cooperation and Development (OECD) principles of governance (see [ECoD99]) in 1999 or the fair information principles set in 1994 in [Rei94].

Privacy by Design seeks to unify privacy methodologies, define processes via a "Privacy Impact Assessment Guidelines" and identify fundamental principles in a holistic framework. The core of the framework comprises seven fundamental principles which stress the importance of considering privacy from the early steps of design and through the whole lifecycle. The core principles are:

Be Proactive not Reactive. Privacy should be included in a system preventively, that means in the design and architecture, not remedial.

Privacy as the Default. It has been shown that the default settings of a system are mostly used in the life-time of a system, even if options are available. Therefore, the most privacy preserving state of a system should be its default.

[4]For the claim, see https://www.ipc.on.ca/images/resources/privacybydesign.pdf, URL last accessed August 17, 2017.

Privacy Embedded into Design. This point re-enforces that privacy should be embedded into the design.

Full Functionality. Integrating privacy and security does not have to reduce the functionality or the utility of the system. There are many technologies to choose from for this purpose.

Full Life Cycle Protection. Data protection should be present for the whole life-cycle of data.

Visibility and Transparency. Visibility and transparency have to be employed to create trust to users. Privacy by Design mentions three concepts: accountability (the responsibility for personal data should be clear and documented), openness (show what data is used and how it is processed) and compliance (necessary steps to monitor, evaluate, and verify compliance with privacy policies and procedures should be present in a system).

Respect for User Privacy. User centric development has been shown to increase the success of products, therefore privacy protection should be regarded as a tool to make a system user-centric.

The Canadian model is considered foundational, but it needs additional frameworks to support system engineers due to its high-levelness, see for example the opinion of [GTD11].

3.2.1.4 PRIPARE

PRIPARE, see [Not15], aims to provide a PbD methodology and process reference model for systematically incorporating PbD in software engineering. PRIPARE has identified a lack of privacy practice that can be used through system engineering lifecycle. Therefore PRIPARE defines a methodology that includes processes and best practices in order to integrate them to system engineering phases (analysis, design, implementation, verification, release, maintenance and decommission).

The PRIPARE approach is goal-oriented and a risk-based. During the PRIPARE process, privacy requirements and goals are defined and integrated to other system goals such as those for functionality and security.

The PRIPARE methodology addresses a system's architecture from a privacy point of view as well as privacy requirements may affect the architecture changes.

PRIPARE defines phases, which are (see [Not15]):

Analysis. In this phase, the functional description of the system is specified and high-level privacy requirements are elicited. The goal of this phase is to understand what privacy controls must be implemented to effectively operationalise privacy in the system.

Design. The design phase focuses on how the privacy controls have to be build. The definition should contain hardware and software architecture, components, modules, interfaces, and data flows.

Implementation. In this phase the system described in the design phase is implemented, following an architectural model and privacy enhancing design principles.

Verification. This phase ensures that the system meets privacy operational requirements. PRIPARE proposes to check implementation properties with formal verification, code reviews and dynamic flow analysis. Furthermore, “posteriori" compliance controls are implemented to support accountability.

Release. This phase defines several processes that have to be completed before system release. The processes include elaboration of an action plan to respond to the discovery of privacy breaches, creation of a system decommission plan, and a final privacy review.

Maintenance. In this phase a data controller has to react to privacy incidents and try to minimize the damage for affected subjects as much as possible. This phase requires immediate actions and a well defined communication plan with subjects and authorities.

Decommission. The purpose of this phase is to correctly dismantle the system according to applicable legislation and policies. The decommissioning of the system should not result in possibilities for data breaches.

PRIPARE has united existing methodologies to accompany a system lifecycle with similar phases to those that are the described in this Chapter albeit with some differences. In Section 3.3, it is explained why such phases are proposed, what their goals are and how they play together with security lifecycle development.

3.3 A Privacy Development Lifecycle

In the previous Sections, several privacy principles and frameworks were presented. In this Section, a privacy development lifecycle is proposed[5] to introduce an privacy engineering systematic where the principles and frameworks can be applied on. For this goal, the processes in a SDL are analysed.

The term SDL or "System (or Software) Development Lifecycle" describes a process for planning, creating, testing, and deploying an information system. More specifically, a security lifecycle development was introduced as a systematic approach for security in software engineering, see [HS05]. A privacy development lifecycle should have similar goals, e.g., systematically introducing a privacy methodology in system engineering. Security and privacy development lifecycles will have significant differences, but if done in an integrated way, beneficial synergies form both approaches can be obtained.

The Microsoft Security Development Lifecycle (SDL) [Cor12b] and the OWASP Software Assurance Maturity Model (SAMM) [CD12] are two of the most popular security development lifecycles primarily for software development. Overall, the seven steps[6] of both security lifecycle frameworks are the following:

- Train personnel or ensure that personnel are qualified.
- Identify threats, evaluate risks (which threat is going to be mitigated, which threats and risks will be simply accepted?) and elicit requirements.
- Design the system according to the requirements.
- Implement the system, fulfilling all requirements.
- Verify if the system fulfils the requirements.
- Deploy the system while making sure the requirements will still apply in the deployment environment.
- Keep the system developers ready to respond to any conflicting or emerging situation.

[5] Note: this Section has been previously published in [AAC16].

[6] The initial steps *Strategy & Metrics and Policy & Compliance* of SAMM are not presented for the sake of simplicity.

3.3.1 Education of System Developers

As with security, training developers in privacy topics is necessary. Although all team members should understand why privacy protection is fundamental and be familiar with the main guiding rules (say, the EU Data Protection Rules or the applicable guidelines), it is assumed that at least one person in the team is particularly well trained in the technical aspects of designing and implementing privacy friendly systems. This person is tentatively called the "privacy expert" of the team. He should know a privacy engineering framework like PRIPARE. The most critical condition for achieving a privacy-friendly product is the presence of one or several privacy experts in the team. The expert is responsible for data protection expertise in the development team and should be consulted in every phase of the lifecycle. He also brings the knowledge where to find mature technical privacy solutions (PETs) and best practices. The lifecycle itself does not focus on developing new technologies, which could cost a considerable amount of time, research and technical expertise, but on using existing building blocks and suitable PETs. A brief overview of PETs for the IoT is given in Chapter 4.

Particularly the Internet of Things requires specialized technology for computational and battery constraints. A privacy expert needs therefore continuous refinement of technical skills and state-of-the-art knowledge.

Legal support will probably be of need to resolve privacy related emergencies. In these situations a privacy expert should be aware when legal support is required.

3.3.2 Phase 1 - Purpose Definition and Data Minimization

The first phase of the privacy lifecycle development is the specification of requirements for the system. Here, the system's functional requirements are analysed by posing the following questions, which follow the principle of *Purpose*: specifically, what personal data does the system collect? What is their specific purpose? Can the system reach its desired functionality with less personal data?

The following process is iterated to stepwise obtain more concrete and operational privacy requirements or PETs.

- Obtain or define the system data flow and the system's functional goals and requirements.
- Determine which personal data is needed to achieve the system's functional goals.
- Analyse the functional requirements and determine if existing PETs can help to minimize the data that is needed for the system.
- Determine the limits of data usage and data retention in the system (say, data is deleted after 2 weeks).
- The privacy expert analyses the proposed solution and suggests new possible technologies to reduce data usage in the system.

3.3.3 Phase 2 - Threats and Risks Evaluation

After the definition of the required personal data in the system, privacy requirements and privacy goals, this phase is used for privacy threat analysis. Several frameworks for privacy threat analysis have been proposed, such as LINDDUN [WSJ15], PriS [KKG08] and FPFSD [SC09].

LINDDUN is especially well-suited for the integration of a privacy development lifecycle as it is based on STRIDE, see [HL06], part of the SDL. System developers trained in SDL should be able to learn the LINDDUN method easily, reuse existing system models, particularly Data Flow Diagrams (DFDs) for their systems and see synergies or problems of both security and privacy goals.

LINDDUN follows STRIDE in defining six steps. The first three cover the "problem space", focusing on the problems, identifying privacy threats and defining requirements of the system. The last three steps cover the "solution space" which aim at fulfilling the requirements, see [WSJ15]:

Define Data Flow Diagrams of the system. In this step a graphical representation of the information flow in the system is created. This step is equal to the step in the STRIDE methodology and could be combined with LINDDUN.

Map privacy threats to DFD elements. In this step system components are

mapped to privacy threats. LINDDUN[7] defines seven threat categories: *Linkability* is the property of linking two or more actions/identities/pieces of information. *Identifiability* is the property of linking the identity and an action or information. *Non-repudiation* is the inability to deny a claim. *Detectability* is the property of being able to distinguish sufficiently whether an entity exists or not, *Information Disclosure* is the property of revealing confidential information. *Unawareness* is the property of not knowing the consequences of sharing information. *Non-compliance* is the property of not being compliant with legislation, regulations, and policies.

Identify threat scenarios. LINDDUN provides so called threat trees to identify threat scenarios. The privacy analyst should examine each of the branches of the tree with a specified DFD element in mind.

Prioritize / analyse risks. All the potential privacy threats that are suggested by the privacy threat trees are evaluated and prioritized via risk assessment.

Elicit mitigation strategies. The suitable mitigation strategy for each threat is determined.

Select Privacy Enhancing Technologies. The classification of privacy enhancing technologies according to the mitigation strategies to which they adhere enables a more focused selection of suitable privacy enhancing solutions.

LINDDUN also supports the integration of any risk assessment framework, for instance the one the security team might in the SDL.

But there is a significant difference: in security, risk assessment is used to prioritize protection mechanisms, to identify high and low risks, and to decide whether a risk is simply taken without mitigation. This is not the case for privacy. Wherever personal information exists in the system, it should be protected in an adequate way. Several steps of this phase are similar in security and privacy and, depending on the frameworks, can be unified. The first step (defining a data flow for a system) can be unified for SDL and LINDDUM, as the same DFD of a system can be used for both, which is a great benefit. In the second step, the threat mapping occurs, followed by the third step of identifying threat scenarios.

[7]The acronym LINDDUN stems from the seven categories.

In this step it becomes clear which components of the system are valuable assets from a privacy and security standpoint. Privacy related assets may have been missed in the security part, thus making this step complementary.

Risk analysis in security is used to decide what countermeasures should be implemented, but a decision might change if again a privacy point of view is added to risk analysis. For example, a component that may have been categorized with low impact from a security perspective might become critical if privacy related risks are high. This could lead to higher investments in privacy and related security protection for that component.

3.3.4 Phase 3 - Design

The design phase develops strategies for implementation, verification, release and response. In this phase also functional, security and privacy requirements are adjusted to one another. For example, functional requirements might need to change to respect policies, security procedures might need to be adapted to support unlinkability and privacy requirements might turn out impractical due to core functional requirements and need to be reshaped.

Conflicts might appear between goals, therefore best-practices can be useful. Best practices are strategies that have been employed by others with good results. For example, Hoepman, see [Hoe14], has defined eight design strategies for privacy which can be realized using privacy patterns (i.e. best practice solutions), namely:

Minimize states that the amount of personal data that is processed should be restricted to the minimal amount possible. This is the most basic privacy design strategy.

Hide states that any personal data, and their interrelationships, should be hidden from plain view.

Separate states that personal data should be processed in a distributed fashion, in separate compartments whenever possible.

Aggregate states that personal data should be processed at the highest level of aggregation and with the least possible detail in which it is (still) useful.[8]

[8]The reader should be warned that the word aggregation is used in the privacy context with different meanings: for instance Solove, see [Sol06], uses the word to describe "*gathering and*

Inform corresponds to the important notion of transparency: Data subjects should be adequately informed whenever personal data is processed.

Control states that data subjects should be provided tools to intervene in the processing of their personal data.

Enforce privacy policies compatible with legal requirements.

Demonstrate requires a data controller to be able to demonstrate compliance with the privacy policy and any applicable legal requirements.

In [Hoe14], Hoepman provides a set of patterns to each strategy to support their technical engineering. As some strategies have rarely been used before, Hoepman points out that new patterns are needed. The reader is referred to the privacy patterns database[9]. An explanation of how patterns work based on a pattern-language for context patterns in the domain of a smart home can be found in [BFH13].

At this point following principles should be added to the ones just mentioned:

Early Application of Policies and Filtering. The processing of personal data should be in the devices under the control of the data subject or if that is not possible, at the earliest point of its generation. This strategy takes advantage of increased processing power in personal devices. Spiekermann et al., see [SC09], describe how this strategy can eliminate the need for data transfer and remote storage, minimizing unwanted secondary data use, compliance with policies and compliance with consented agreements.

Do not link. Hoepman describes a separation strategy to process data in a distributed fashion, but e.g. storing data in separated databases is not enough, if it can be re-linked across the databases. This strategy helps to avoid such cases by establishing a mechanism that actively checks for possible identifiers and allows proper separation without the possibility to re-aggregate the data.

Usability and Data-flow transparency. Ensuring the usability in privacy controls has several objectives: make privacy controls usable for a variety of

combination of various pieces of data about a person" and in that sense aggregation is a privacy threat. Here it means abstraction or replacement by more general statistics on the data in order to favour privacy.

[9] http://privacypatterns.org

users with different skill levels, integrate privacy controls seamlessly into the system and make the users understand what they are seeing and what they can invoke with the controls provided to them.

Rubinstein et al. propose in [RG13] to use field studies, interviews, surveys, and related methods to understand the user requirements, pain points, and expectations, for the creation of narratives that help drive software engineering requirements, which are then incorporated into the overall development plan. The narratives or scenarios should be transparent to the user, allowing him to visualize how his personal data is used and how it flows in the system.

3.3.5 Phase 4 - Implementation

Proper documentation and by-default configuration are keys in this phase. Users must be able to perform informed decisions about their privacy without much trouble. In other words, the system should behave privacy-friendly out of the box. This is called "privacy-by-default" and is one of the most important fair information and privacy by design principles, as the majority of users will interact with a system in its lifetime with the default settings, as pointed out in [Wil14].

"Secure Coding" procedures will be needed to avoid privacy issues, which could otherwise become visible later in the system. PETs need to be securely implemented in the same way as security mechanisms, e.g., by coding experts and verified with code reviews. Implementation strategies, as defined in phase 3, help to assure that the implementation effort is controllable, timely and reaches the desired quality.

Software developers and privacy experts should work closely in this phase to avoid problems such as an improper choice of libraries with unwanted effects (like the use of logging of data including personal information, the presence of vulnerabilities or leaks).

3.3.6 Phase 5 - Verification

Verification is an important process in a security lifecycle. It serves different purposes: to test the used security mechanisms and to evaluate the cost-benefit of those mechanisms.

It remains unclear if security testing procedures, such as pen-testing, fuzzing, etc., can be used for privacy purposes.

But the methods used in code review offer also good insight into data, the information flow in programs and about the presence and enforcement of privacy-enhancing mechanisms.

Also, specific test based on statistical analysis and machine learning could help to evaluate privacy enhancing technologies.

3.3.7 Phase 6 - Release of System and Education of Stakeholders

Phase 6 is used to develop strategies in case that vulnerabilities are discovered on release. These strategies are carried over to the next phase [Cor12b]. Strategies cover assignment of responsibilities, emergency response methods and emergency assessments, technical actions and communication strategies.

Privacy cannot be protected simply by technical components and this holds for security as well. The education of system stakeholders takes a significant role in this phase. Stakeholders of the system are system administrators, operators and system end-users. Operational stakeholders need to know which data is processed by the system and what kind of implications this might have for users. Technical protection might be useless in certain scenarios that might seem unlikely, yet the operators should know them to be able to react in case they occur.

Data subjects need to be informed about how their data is processed and which tools are provided to exercise their privacy rights. The released system should be accompanied by an according privacy disclosure which describes the system's use of personal data, by a documentation of privacy tools that the system provides and user communication tools like a "quick" or Frequently Asked Questions (FAQ) text to addresses likely user questions.

Users, and in particular system administrators and other personnel that may interact with the system, need to be informed about how their actions affect the privacy of others and which actions can lead to privacy violations.

SDL proposes to validate the system's privacy standards by a privacy advisor or a privacy seal of quality[10] prior to release. A legal privacy expert should review the documents and overview the release process.

3.3.8 Phase 7 - Response

The last phase is one of the most significant in the lifecycle. It uses the results from the release phase for rapid response strategies.

Breaches might have a significant impact, as discussed in Chapter 3, therefore the team must be prepared to respond efficiently and timely to them as they can occur unexpectedly. A response team must therefore develop a response plan that includes preparations for potential post-release emergencies. The Canadian Office of the Privacy Commissioner (OPC) proposes in [Can] four steps for this phase:

Breach containment and preliminary assessment. In this step immediate actions to stop the breach are carried out with an assigned investigation leader and a response team. Legal action against the attackers is suggested as well.

Evaluation of the risks associated with the breach. In this step the risks associated with the breach are evaluated and first actions are triggered. The risk depends on the amount, sensitivity and context of the compromised data, e.g., if the data was encrypted or not and if identifiers or other information links them to particulars. Assessments can help to identify the individuals affected, the root-cause and the foreseen harm and find adequate mitigation strategies.

Notification. In this step the users are notified of the possible consequences the breach might have. The notification should be as soon as (reasonably) possible and personal, by phone, email etc. In this step, also further

[10] For example, a privacy seal of quality can be obtained in Germany by ULD, see https://www.datenschutzzentrum.de/guetesiegel/

organizations can be informed, such as cyber-defence centres, credit card companies (if credit card data was stolen), etc.

Prevention. A prevention plan is defined. The OPC suggests the level of effort should reflect whether it was a systemic breach or an isolated instance.

This steps aim at fast communication and support strategies between companies and users. They help the users to understand what possible consequences the breach may have and give them a transparent view of the emergency response strategies form the company.

A legal support might be needed to handle consequences, but also to initiate legal actions against the attackers. A root-cause mitigation team investigates why a breach was possible and develops a mitigation plan that has to be realized rapidly.

3.3.9 Challenges of Privacy Engineering in IoT

The challenges of privacy engineering in IoT are based on the structural problems of privacy engineering for any kind of ICT system. The characteristics of IoT systems, as described in Section 2.2, bring additional challenges for privacy engineering.

3.3.9.1 Best Practices for Privacy Engineering

The proposed lifecycle of this thesis in Section 3.3 covers the most cited and reasonable proposals from the scientific community for engineering privacy. In order to measure the performance of such a lifecyle, the security lifecycle frameworks can be used again as a comparison. A Security Development Lifecycle (SDLC) relies on security standards and best practices driven by the interest from research communities, organisations and companies. For example, the Microsoft Security Lifecycle which has been proposed in 2004 and has been a mandatory practice since, see [Cor12a]. The security is coupled to the product lifecycle management processes through policies, such that every developed product has to comply with the security policies.

Privacy engineering has neither been the focus of organizational interest and nor has it been developed from a business perspective. Privacy is driven by

law enforcement and has thus been treated as such in corporate environments. Corporal experience exist in jurisdictional interpretation of the privacy protection directives and not in the product lifecycle integration and the technical engineering of privacy.

A privacy development lifecycle needs to be adapted to corporal and organizational needs. For example, the evaluation of market data and needed resources to start a project is a typical process in product lifecycle management. If the privacy requirements are not identified in the same phase, the successive decisions on resources for training developers, the elicitation of requirements and the development of technical means may be ill-advised.

The new privacy protection rules as presented in Section 3.2.1.2 motivate companies to do more in terms of preparation and integration in other lifecycle processes. The alignment of privacy, portfolio and product lifecycle requires privacy experts as well as project leaders, project managers and quality managers. Their interactions will have to be adapted per company and organization and will need time to materialize to measurable best practices.

3.3.9.2 Cognitive and Structural Problems

The two foundational principles of privacy are centred around the transparency of privacy and privacy self management. The first principle of *consent* describes acquiring an agreement on the collection, processing and disclosure of data between a data subject and the processing party. The second principle of *purpose definition* demands the formulation of the purpose the data is collected for, processed and disclosed.

In practice, these principles are implemented in the form of privacy policies. Privacy policies are usually a unilateral proposal (from the service provider to the user) on the purpose of data collection, use and disclosure. The data subject is normally a user that wants to access an ICT system and who has to give his consent on that privacy policy. Until now, consent has rarely been demanded explicitly[11]. When the user starts using the system, he has implicitly given his consent.

[11]The GDPR requires consent to be demanded explicitly.

Solove has analysed in [Sol12] these foundational principles from the point of view of practicability and has identified several cognitive and structural problems that hinder the proper application of both principles.

Cognitive Problems. Privacy protection is regarded as a compliance requirement. Therefore privacy notifications are often formulated with the intention to comply with those requirements in contrast to their original purpose of notifying the users and to inform them about what will happen to their data. In consequence privacy policies are often covering requirements of privacy regulations in long, hardly legible text fragments.

According to Solove, legibility is only one cognitive problem. In general, users are uninformed of why privacy policies exist and what their context is. Uninformed users will access the system without reading privacy policies and even when forced to, they will skip through as the purpose of the policies itself remains unclear.

Legibility becomes a problem when users want to exercise their rights and read the policies, but ultimately give up on understanding the formal, legal statements that are described in them. Solove states that even when users understand the notifications, they often lack the background knowledge to make an informed choice as whether to consent to those policies or not.

The reader is reminded of the Schrems vs. Facebook case, where a highly educated law student was able to recognize incoherency in the privacy policies and the practices of Facebook. Schrems is a public example of users that can understand the privacy notifications, understand the implications of the policies. But even if this group of users exist, their decision can be skewed in various ways. For example, a service can suggest to be available at a special rate only for a limited time and draw the user in consenting in favour of economical value. Often, services offer a "all-or-nothing" deal, where consent rejection leads to the rejection of the whole service[12].

In the Internet of Things, service provision may be highly distributed among many subcontractors in several countries. The cognitive problems described by

[12]The reader is reminded of the introductory quote: *"Necessity is blind until it becomes conscious."*

Solove aggravate with every additional layer of providers. Providers maybe elicited dynamically, thus needing a on-demand notice on use, processing and disclosure.

IoT systems may also affect non-users, as described in Section 2.2. The notification of non-users or specifically, their identification and their execution is an additional challenge for privacy self-management. Also, the description and evaluation of multi-party consent in privacy policies is a novelty that cannot be represented by static, unilateral privacy policies made by the service provider.

In this thesis consent has been evaluated through a central management system. A proposal for a consent management system is presented in Section 4.2.2, although the statement in Section 3.2.1.2 still holds: multiple-party consent can only be clarified by new regulatory guidelines that consider ownership and possession[13].

Structural Problems. Solove further shows in [Sol12] that cognitive challenges can be generalized to show that consent and purpose mechanisms have structural problems as a consequence. This problems again aggravate for the Interent of Things.

Firstly, consent does not scale well. The cognitive problems described above apply per service provision or ICT system. In the case of IoT, where several services and applications might request a user's consent for each service, service composition or similar, the user will be faced with one or several possibly very complex privacy policies. Technical solutions could help, such as a consent management assistant that supports the user to automatically reject or accept certain pre-defined purposes based on user defined policies. Prerequisites are standardised, machine readable privacy policies which are able to represent several layers of data processing. Some privacy policies have been proposed to be machine readable, see [Cra03], but no format has achieved general consensus, therefore machine readable policies remain a future challenge.

Secondly, consent mechanisms in the form of privacy notices do not aggregate well. Data collected by one IoT system maybe aggregated with data of many other systems to reveal information that was not seen before. The aggregation

[13]The reader is reminded of the problem with an example: a subject enters the car of another subject, where the car senses the comfort quality of the passengers by sensing their heart rate, transpiration, etc. If the owner of the car consents to the evaluation but the guest does not, how is the conflict solved?

may reveal also new personal information of subjects that has not been consented to by the respective person. Here, additional privacy principles like intervention, transparency and access can help to inform the subject about the new information that has been retrieved, even if the user has not consented to it. However, it remains to be evaluated how a technical implementation performs when several applications with respective subcontractors are in place.

Finally, the problem of assessing the harm of the disclosure of personal data centres as the main problem that leads also to many of the cognitive problems. Solove underlines privacy protection is the management of personal data over a long term. The possible effects are perceived to be none, if not immediately obvious (e.g. the consequence of the disclosure of embarrassing pictures). Users have to decide over privacy individually and far in advance, making the need for transparency more critical.

Next Steps. The cognitive and structural problems show that privacy self-management based on consent is problematic. Solove notes that consent is an undertheorized concept, see [Sol12] and proposes therefore four complementary directions:

Rethinking Consent. Solove formulates a proposal for law enforcement, where consent is not validated in a binary way (e.g., consent has been given or not), but as a concept with many nuances.

Developing Partial Privacy Self-Management. Solove argues that privacy self-management is needed for users that want to use privacy related services (such as social networks) and those that don't want to. Solove sees a similarity between privacy aspects and safety aspects. Users have a wide range of freedom when buying goods that require safety (e.g. cars or food) but those goods are regulated, such that micromanagement of risks by users is not needed.

Adjusting Privacy's Timing and Focus. Consent and purpose-binding are concepts that target the initial relation between users and service providers. But data may reveal more information about a user in a future point of the service provision where data analysis or aggregation took place. Solove calls

this the timing of privacy protection. He proposes that consent should not be asked for once and be valid in advance, but it should be required when the new data is becoming visible.

Moving Toward Substance over Neutrality. Solove underlines that consent can be used to "*[waive] many constitutional rights*". Consent can be used to equally accept many forms of risk. Solove proposes to use a form to codification similar to the Uniform Commercial Code (UCC) to rate risk more precisely and to base consent on it. The UCC categorizes different responsibilities and risks in sales and commercial transactions, see [Hil76].

The proposals by Solove can be directly transferred to privacy engineering in IoT. The application of consent has been targeted by privacy enhancing technologies in the form of policies, where privacy policies and privacy agents try to capture the nuances of consent in languages like Extensible Access Control Markup Language (XACML). A proposal for the architectural integration privacy policies in IoT is presented in Section 4.2.4. The problem with XACML and similar policy definition languages is that the languages are complex and heterogeneous, thus often fail to find practice. A legislative definition of fine grained consent could help as a foundation for further refinement and a common understanding of privacy policies.

Privacy agents target partial self-management. Agents act in the name of the user and constantly monitor the privacy requests and data flows of the user. The agent's decisions are based on the user's policies and inquire the action of the user only when policies are unclear, a technical proposal is given in the "activator/deactivator of data collection", see [RER15]. Evidently, agents act according to the user's decisions. If the user is unaware of certain risks, the agents will not prevent him from miss-assessing the risk. Technical and legal solutions to partial privacy self-management are therefore complimentary and cannot advance without the other.

Timing of privacy has direct relevance to IoT. Data aggregation by a service provider is a focus topic of IoT. Consequently, the renewal of consent at the time of privacy related revelation of information is a duty of the service provider and can be introduced and motivated by law and compliance.

Solove does not go into technical details, but as formulated above, for most regulatory foundations, respective technical representations are needed. As mentioned in Section 2.19, IoT has several constraints where traditional privacy enhancing technologies cannot be used. In turn, that means that even with new regulation, privacy could not be engineered into systems because required technology is not present.

In this thesis several technologies have been evaluated as a proof of concept to allow privacy enhanced technologies in IoT. The constraints have been introduced in Section 2.6.1, the technologies are based on the use cases of Rerum, see [RER14a] and are detailed in Chapter 4.

Chapter 4

Privacy Enhancing Technologies for the Internet of Things

Chapter 3 firstly mentioned the need for privacy enhancing technologies in the General Data Protection Regulation and in the proposed privacy development lifecylce, see Section 3.3.

This Chapter introduces several technologies based on the requirements of the Rerum use cases[1].

The technologies are categorized according to Güerses et al. proposal, see [Gür14], namely privacy technologies for *control, practice and confidentiality*. The technologies serve as a proof of concept. Their elicitation, development and evaluation aimed at the implementation of the Rerum trial use cases and follow the same constraints as described in Section 2.6.1. The Rerum use cases can be found in [RER14a], they comprise UC-O1 Smart Transportation, UC-O2 Environmental Monitoring, UC-I1 Home Energy Management and UC-I2 Comfort Quality Management. Their economical background is described in Section 2.1.

[1]Note: the content of this Chapter has been previously published in [SWC+15] before. Some of these technologies have become intellectual property of Siemens AG, an additional note will be given in the respective Section.

4.1 Categorization of Privacy Enhancing Technologies

The technologies presented in this thesis are categorized according to Gürses work in [Gür14] and according to the "hard" and "soft" privacy control definition of [SWC+15].

Gürses describes the three categories of privacy research as follows:

Privacy as Confidentiality. Gürses characterizes privacy as confidentiality with three principles, *data minimization, avoidance of a single point of failure and openness to scrutiny.* Data minimization enhances privacy by minimizing the acquisition information. Avoidance of a single point of failure means an architectural decision to avoid any single point of data acquisition within an ICT system. Openness to scrutiny denotes the openness of the design of PETs to the public eye in order to increase the maturation and the trust in the respective technology. A wide known technology in this regard is the TOR network, see [MBG+08].

Privacy as Control. This type of privacy research supports methods to inform users about the purpose for which they are consenting personal data collection, which data is exactly collected and the period the data is stored. Related technologies are access control mechanisms, policies and dashboards.

Privacy as Practice. This research category analyses the mediation between transparency and feedback mechanisms in IT systems, and privacy related decisions of users. The central assumption is that the higher the privacy awareness is of a user and the higher the feedback is of a system to a user, the better is the user's decisions concerning his privacy. One example comes from online social networks where a user might or might not post an image if he realizes that it will be publicly visible.

Hard and soft privacy controls are categorized as follows:

Hard privacy controls. Hard privacy mechanisms enforce privacy as confidentiality and privacy as control with technical means. The mechanisms are verifiable and are often under the control of the data subject (e.g. the user). Such mechanisms can provide data minimization (e.g. reduce granularity

of data), anonymization (hide a user's identity) and unlinkability (several actions of one user are not linkable by a thrid party) among other concepts.

Soft privacy controls. In Gürses description of privacy as confidentiality, the second principle detailed the avoidance of a single point of failure, i.e., the avoidance of a single point of data storage and the "trust" that this point confiably protects that data. If this principle cannot be achieved due to some constraint (e.g. scenario specific), controls are applied that are denominated as soft privacy controls. That means, that the controls cannot be verified or enforced, but they are merely a supporting mechanism for the data controller. One example are sticky policies that travel with data of a user and state under which circumstances the data is allowed to be processed. In this case, the policies cannot be technically enforced, the data controller is assumed to be trusted in following the policies.

4.2 Privacy Enhancing Technologies Supporting Practice and Control

Diaz and Gürses describe in [DG12] privacy enhancing technologies supporting practice as technologies that raise transparency and understanding about the flow of personal data through feedback and awareness. If users are able to understand how information is collected, aggregated, analysed and used for decision or value creation, they might be inclined to question, intervene, and renegotiate their decisions regarding their personal data.

Privacy supporting control is described in [DG12] as technologies that provide a means to users in order to control the disclosure of their information and at the same time gives some framework for service providers to adequately define privacy policies. The technologies should additionally enforce those policies in a way that prevents the abuse of personal information for illegitimate purposes.

The technologies presented in this Section support Rerum use cases UC-O1, UC-I1 and UC-I2 [SWC+15] and represent both aspects, practice and control. Although each technology *could* be classified into either the control or practice paradigm, the usefulness of each technology only becomes evident with the complement of

both paradigms. An example: Diaz and Gürses categorize access control and dashboard mechanisms in the respective paradigms control and practice. On the one hand access control mechanism cannot be applied without somehow user-friendly interface for the definition of policies, something that is provided in the Rerum privacy dashboard, a single point of contact for the user. On the other hand, the dashboard (or privacy mirror) misses the underlying components that allow to make data flows transparent to the user. Gürses [Gür14] notes that users can be encouraged opportunistically to review their privacy settings with transparency mechanisms. For this practical reason, the paradigms of practice (review privacy settings) and control (edit privacy setting) will be contemplated as one.

The technologies in this Section will be presented individually. In Section 5 all technologies are integrated into a final IoT domain model.

4.2.1 Privacy Dashboard

The intent of a privacy dashboard[2] is to help users gain an overview of the personal information collected about them, particularly when the data sources, personal data and related services in question are as numerous and unobtrusive, as in IoT. This is achieved by a graphical interface which enables the management of policies and consent for the private information of users [ZAM14].

The dashboard supports the principles of *individual participation*, *notice* and *access*, see Section 3.2.1.1, fulfils the requirements to Right of Access of the GDPR in Article 15 [Par14], and is classified as a transparency enhancing tool, see [JWV13]. A privacy dashboard answers the question "what does a system know about the user?".

Since it is impossible to expect that all users of an IoT system have a strong technical background, it is not viable for them to express their privacy policies in a complex policy language. The graphical interface visualizes how a user's devices behave in an IoT system and it allows changing that behaviour according to the user's preferences. These preferences are converted (automatically) to policies in a machine-readable format. Additionally, the privacy dashboard helps to understand

[2]The content of this Section was previously published in [SWC+15, SC16].

what kind of data their devices are sharing with the IoT system[3], thus avoiding that users are overwhelmed with raw data.

4.2.1.1 Related Work

The idea of privacy dashboards exists and has been practised successfully. Most prominent is the example of Google, which provides a dashboard for users to overview the data linked to their accounts and to remove or edit their interests via an option called "ads settings" [Inc16]. Rubinstein and Good note that is now commonplace for ICT services to provide users some form of access to their personal data, see [RG13]. The idea of the privacy dashboard has also been described as a privacy pattern, see [DGZM15].

The first common privacy vocabulary for machine readable privacy policies was proposed by P3P, see [Cra02]. This vocabulary has been the foundation for many dashboard implementations. Doty et al. have described the privacy dashboard as a privacy pattern.

Janic et al. categorize in [JWV13] implementations of dashboards and transparency enhancing tools in four categories:

1. Tools that provide insight in intended data collection and processing based on website privacy policy. A dashboard implementation for web browsers has been provided by the W3C in [Rag11]. Costante et al. propose a graphical representation of website privacy policies in [CSPH12]. Implementations for web browser exist as seen in Privacy Icons [OPW+12] and Privacy Bird [Cra09].
2. Tools that provide insight in already collected and/or stored data. The European projects PRIME and PRIME Life have developed the Data Track Privacy Dashboard [WS10], a client-side tool that tracks the information that users disclose to websites by intercepting his traffic. Google Dashbaord is a server-side example of tools in this category, it shows what data has been collected and processed by Google's services.
3. Tools that provide insight in third parties tracking the user. Some examples

[3]It is assumed that the user knows which devices he owns. This might be based on the ownership model presented in Section 2.5.3.3.

are provided by Angulo et al. in [AFHPW15] and by the Mozilla Firefox plug-in Lightbeam, see [VKC15].

4. Tools that provide insight in data collection and processing based on website's reputation. The most famous tool is the Web of Trust[4] [WOT07].
5. Tools that raise awareness of possibly unwanted data disclosure by promoting awareness. Some examples are Friend Inspector and Privacy Score for Facebook[5]. Me and My Shadow [Tec12] is a website that helps users to understand privacy risks, how technology of tracing works and gives practical advice on how to better protect privacy by using PETs and by changing common behaviour.

Janic et al. come to the conclusion that although all of the tools support awareness, none of the tools show the user where his data is stored and how it is processed, merely which data he has released to the public.

Fischer-Hübner et al. have proposed in [FHAP13] an approach towards tracking user data throughout service provision. The approach relates to the Data Tracker [WS10] and extends it with further logging capabilities. The created logs are based on a multilayer of different policies throughout service provision and show a trace of the services that have received the user's data. The logs contain respective data storages, processing and disclosing policies used within the traces. The approach takes usability aspects into consideration and addresses ex ante (which information does the user need to give his consent?) and ex post transparency (what does the system know about the user?).

Bier et al. propose in [BKB16] "Privacy Insight", a dashboard based on data flow and data usage control, as defined in [PLB12]. The dashboard is deployed by the service provider in a technology independent, user friendly way (e.g. by showing tool-tips) and is routed into an XACML policy architecture. The dash board is supported by event listeners of operating systems and tracking components. Event listeners intercept system events and forward them to the

[4]The Web of Trust service has received a considerable amount of critique as the WoT service seems to acquire personal data of its users and sell it. Additionally, the reputation system is supposed to be affected by fraud. See https://www.kuketz-blog.de/wot-addon-wie-ein-browser-addon-seine-nutzer-ausspaeht/, last accessed on August 17, 2017, for a snapshot of the discussion (German).

[5]Both tools have been referenced in literature but were unavailable as of August 17, 2017.

tracking components. Event listeners are coupled with a policy enforcement point and the tracking components are interleaved with a policy information point to allow the information flow to be checked by the tracking component and to enforce the user's preferences by the XACML architecture.

4.2.1.2 Rerum Dashboard

The Rerum dashboard is a part of the Rerum middleware functional components for privacy, see seciton 2.5.4.

The dashboard offers a graphical user interface to the users, it tracks how many devices the user owns, it visualises for which services the user has given his consent, what data the user's devices have published, and it allows him to define policies based on a simplified language as well as instantly suspend his consent and deactivate data collection. The dashboard is supported by additional Rerum middleware Rerum Security, Privacy and Trust Components (SPT) components: the consent manager, see Section 4.2.2, an extended XACML architecture for the Rerum domain model, see Section 4.2.4 and the activator and deactivator of data collection that is interleaved to the data and context manager, see [SWC+15].

The dashboard follows a similar concept to that of Bier et al. [BKB16], albeit published one year prior, see [SWC+15]. Differently than the Privacy Insight dashboard, the Rerum dashboard is an architectural element and has not been implemented, although the supporting elements have been implemented for the Rerum trials. Also, some design decisions were made to integrate the Rerum dashboard in the Rerum domain model. For example the data provenance model is much simpler as third party service providers were left out of scope in Rerum. This could be remediated by including data usage control and compliance of third party service providers as seen in Bier et al.'s approach.

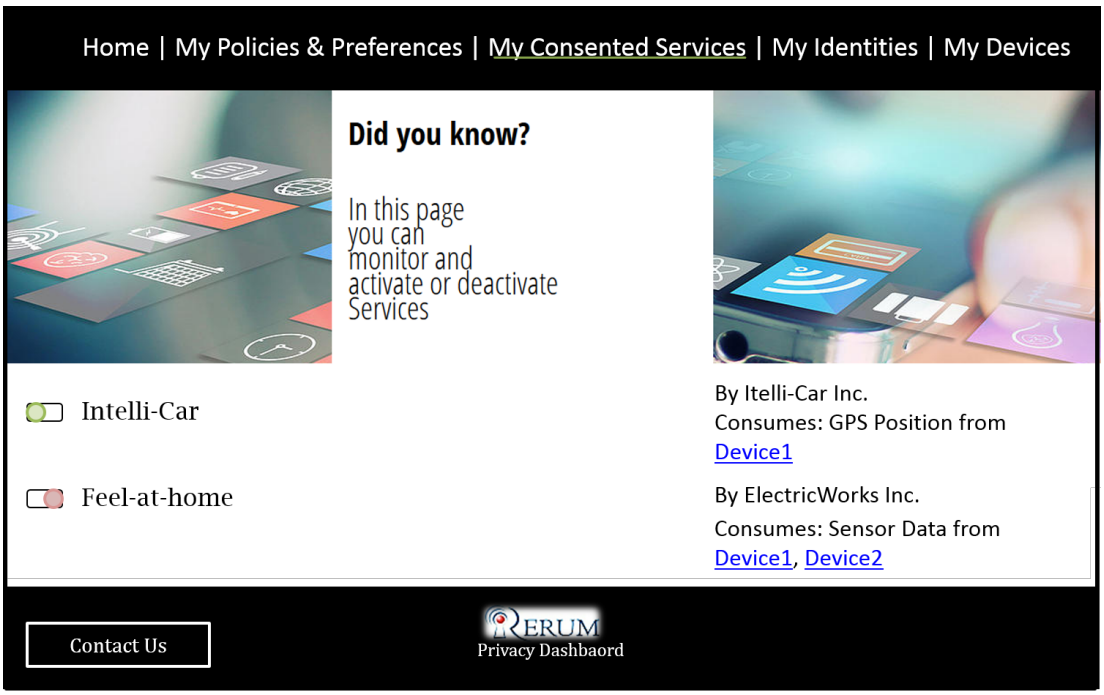

Figure 4.1: Draft for the Rerum dashboard [SWC+15]

Figure 4.1 proposes a graphical interface of the dashboard. A dashboard layout was not specified in Rerum [SWC+15] as Rerum did not target the user application layer. The dashboard could follow the privacy icon proposals form Haduong et al. [HTQ12], Holtz et al. [HZH11] or [Me16] while the mash-up of content could follow the proposal for a dashboard in smart homes by Bush et al. [BKS14].

4.2.1.3 Integration in the Rerum Domain Model

The integration of the dashboard will be detailed based on an example: a service requests access to user data. The consent manager searches for existing policies to verify if consent exists. If no consent exists, the consent manager requests consent from the dashboard. The dashboard visualizes the request, notifies the user and presents a readable form to the user. If the user accepts, the privacy dashboard will confirm the given consent to the consent manager, which will trigger a policy generation and storage. The dashboard will also handle the interaction between a user and his devices. As devices will adopt different pseudonyms, the dashboard will interact with the anonymising / pseudonymising manager to retrieve the real identities, in order to visualize activities to the user. Figure 4.2 depicts the interaction of dashboard and consent manager.

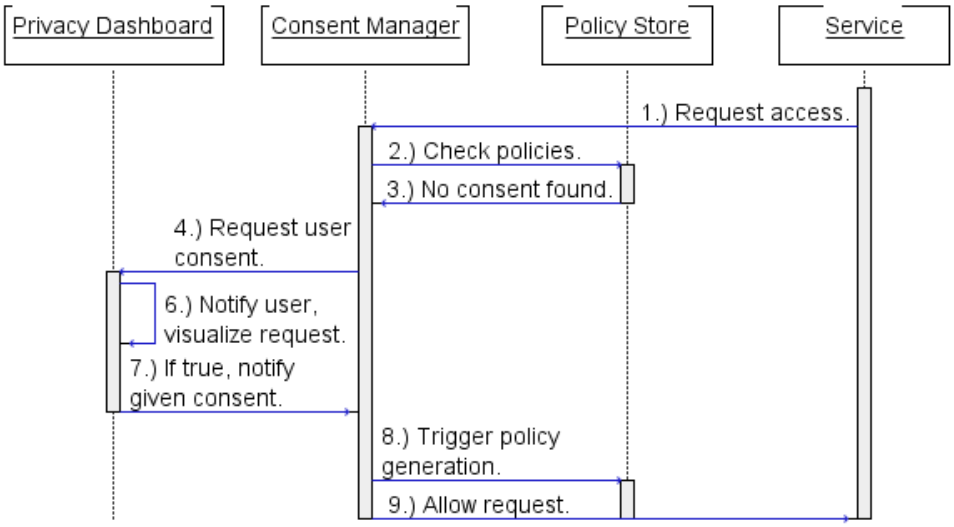

Figure 4.2: Interaction of Privacy Dashboard and Consent Manager

Figure 4.3 includes the Anon./Psnym. Manager to the message flow. The sequence unifies for brevity the Rerum middleware and the service provider without detailing the actions between both.

As a first step, a device agrees on a pseudonym. The anonymising and pseudonymising manager takes care of the pseudonym generation and issues either a pseudonym seed, so the device can further generate more pseudonyms, or a single pseudonym to the device. This agreement (and the mapping of pseudonyms to devices) will be displayed to the user on the dashboard. The device can now identify and register itself under the pseudonym *nym-1* (message 3). Note that the device can register itself under different pseudonyms at the same time. Once the middleware accepted the registration (message 4), it will forward the activities of the device to the respective privacy dashboard (message 5). The relation between device and privacy dashboard is negotiated through the device's ownership and is part of the registration information of message 3.

The privacy dashboard will ask for the real identity of nym-1 and, if ownership relations apply, receive the original identity (message 6 and 7). The privacy dashboard will then present the activities of device to the user (message 8), even if the device uses several pseudonyms.

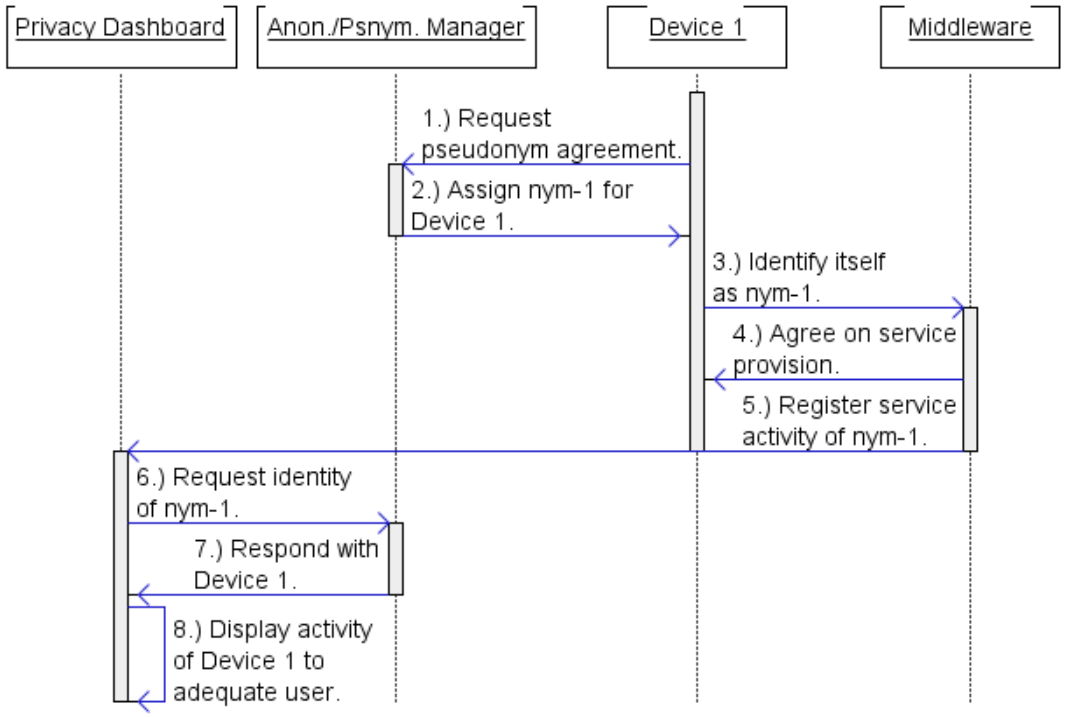

Figure 4.3: Interaction of Privacy Dashboard and Anonymising and Pseudonymising Manager

The dashboard is the main interface for the user in terms of privacy control, a further description is given in the privacy dashboard technical Section below. The user is allowed to interfere anytime, e.g. in case he wants to opt-out from a service. The responsible component is the activator / deactivator of data collection with the privacy dashboard as its interface. Service agreements and device's activities will be shown in the dashboard. The user will have a possibility to opt-out of the service with the click of a button, sending a command to the activator / deactivator and triggering a data collection stop at the Rerum middleware. A detailed description of the activator / deactivator including all its dependencies of the Rerum middleware is given in [SWC+15].

Figure 4.4 shows the sequence of deactivation of data collection after an initial request was allowed.

The example assumes that a request was made to a device and that policies are checked to evaluate if the request should be allowed or not: first preferences of a user policy are matched to the request (messages 1 and 2). The request is allowed and the decision is served to the device (message 3). The device notifies the Rerum middleware of an accepted request and publishes data (message 4).

The service provider requests data from the device through the Rerum middleware (message 5). The request is redirected to the device (message 6) and responded (by the device to the middleware) with a data stream (message 7). The data is redirected from the middleware to the service provider accordingly (message 8).

Simultaneously, the data provision is registered by the middleware to the privacy dashboard (via the activator / deactivator of data collection, message 9). The user may stop the data collection at any time (message 10). If so, the activator / deactivator sends a notice to stop the data collection to the middleware (message 11), note that also a new policy entry could be added for that service. While data is still served from the device and the service provider still awaits or is subscribed to the Rerum middleware for the data stream, the middleware will respond with an inexplicit "Data Stream not available" notice, without publishing any reason. This ensures that the user can intervene in the processing of his data as required by the GDPR anytime without having to fear consequences of the service provider.

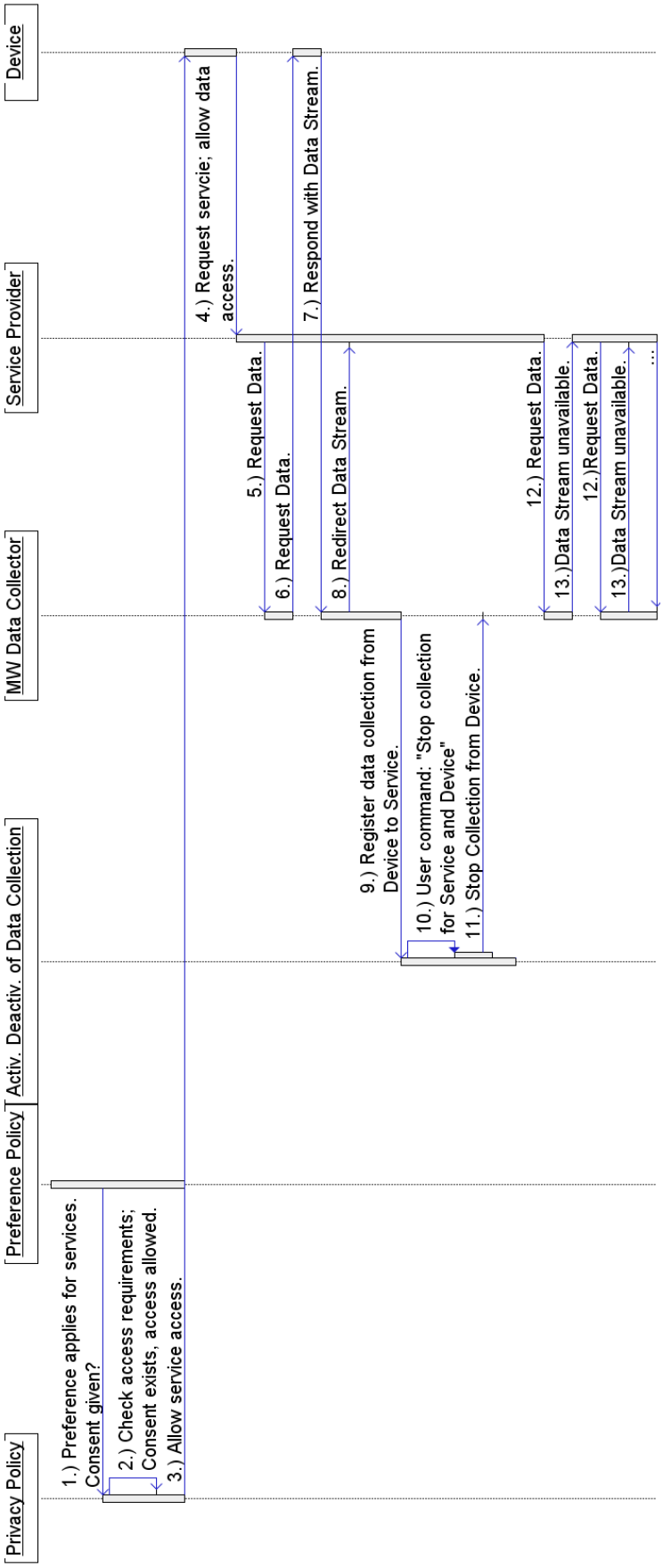

Figure 4.4: Interaction of Privacy Dashboard and Activator / Deactivator of Data Collection

4.2.2 Consent Manager

A Consent Manager[6] is a dedicated component of an ICT system that allows a user to see which applications request access to his personal data, the purpose of the request and the circumstances surrounding the request. A given or denied consent is stored in a consent database, thus retrievable for accountability issues. In addition, the consent manager could help the user to assess the risk and benefit of a request, hence assisting the service provider in offering the user *informed consent* [FLM05].

A data subject may declare his consent to a service provider or withdraw a previous one at any time, thus supporting Consent, Notice and Access, Participation & Accountability. The GDPR additionally demands that "it shall be as easy to withdraw consent as to give it", Article 7(3) [Par14], thus the requirement to offer the user an easy way to manage his consent.

The consent manager is an interesting component for IoT: the user might constantly be requested to consent to services that want to collect data from his devices and offer him some value in return. The user will be able to accommodate this situation by specifying policies which support the automation the consent process. The user must be able to comprehend which services he consented to and under which circumstances.

The consent manager is therefore related to an authorization engine and a dashboard where the final decision on a conflicting request or the withdrawal of it is made by the user personally.

4.2.2.1 Related Work

Related work on the consent manager targets informed consent and a consent manager. Related work on informed consent explicates what is needed in order to inform a user adequately and in such a away that he understands the complexity of the system. The consent manager supports the user in weighting the sensitivity and the risks stemming from his consent against the foreseen benefits.

Related work on the consent manager is the delineation of a technical and/or architectural component that assists the provision of informed consent.

[6]Partial content of this Section was previously published in [RER15].

Informed Consent. For the sake of completeness, the reader is referred tp the consent discussion in Rerum [SWC+15] and the structure proposal of informed consent by Friedman et al. [FLM05]. The most significant insights into informed consent have been provided in the area of Health Care (HC). Researchers in the HC area have mentioned many of the shortcomings of consent that have been subject to recent privacy discussion (a.o. simple language, visualisation, risk assessment support, standard templates, see [SWC+15]). Informed consent has been identified as challenge to explain a patient[7] how a complex system (his body) works and which risks and benefits his consent (and accordingly, the treatment on his body) might have. The same applies for ICT and IoT, a complex system has to be explained to the user in order for him to understand the benefits and risks. The form and methods of explanation might be directly transferred from health care: Cassileth [CZSSM80] surveys in 1980 why informed consent misses the point of informing the user by analysing 200 patients. The methods used (oral and consent forms) were examined and new methods that improve legibility and comprehension were proposed. Appelbaum [ALM87] describes in 1987 informed consent from a clinical and legal point of view, how it affects patients in clinical practice and how the successfulness should be assessed.

More recently tools have been proposed to help the provider to assess the risk first before it is communicated to the patient. In [BLP+13] Biliomoria proposes a surgical risk calculator. The calculator is based on pre-defined clinical data of several clinical institutions. The tool was evaluated positively in Biliomoria's survey. No such tool exists to help with the evaluation of risks for ICT / IoT based systems, but might be a practical approach if the experience of experts is integrated[8].

Consent Manager. An early proposal for a consent management system for web-service environments has been published in a patent by Dunn [Dun06]. The manager is interleaved with an access control component to support the user in

[7] Patient is a user in IoT and other ICT systems

[8] The critical reader may question the need for such tools, as health care services carry much higher risks compared to ICT-based systems. This might only be the case for treatments that directly threaten the life of the patient, but not for others. Privacy breaches might cause significant damage to the affected subjects as well, see [Sch78].

handling several consent requests. Whenever a request is not authorized, the consent management system is invoked. The validation of the request is based on the P3P language [CW07]. The user defines his preferences using the P3P preference exchange language APPEL [Con02] which are compared to the P3P policy definitions of the service and the user is presented one or more options to consent to the service or not.

Other proposals for consent management have been made recently in the area of health care. Researchers in the area of health care have mentioned many of the shortcomings of consent that have been subject to recent privacy discussion.

Dunn's proposal [Dun06] has strong similarities with the Rerum consent manager (matching of policies and preferences, storing of consent, informing the user, interleave with access control), except for the architectural integration. Dunn considers web-services as its main use case, whereas Rerum focuses on use cases with constrained devices. The architectural integration separates both proposals, although being conceptually close.

Wang and Hongxia [WJ12] propose a consent management system based on an own informed consent model for HC. The consent manager supports weighing benefits and risks of a consent request based on *expected benefit, sensitivity and relevance*. Benefits maybe the treatment results of a primary physician (high) or targeted advertisement from a drug store (low). The risks are rated according to sensitivity (generally health status) and relevance, that means which data is requested and how it relates to a service (if irrelevant data is requested for a service, the risk becomes higher). Relevance is rated through statical learning methods. That means that similar requests are compared over time to learn the normal amount of data records that are needed for a special request. The manager also needs a pre-access definition by the patient and an administrator to rate how sensitive records are and how important a request may be.

The architecture of the consent manager is as follows: every request to a patient's health records is redirected to the consent manager. Based on the statistical learning engine, requests are rated and a decision (accept, deny) is suggested to the patient. On acceptance requests are accepted and responded to accordingly. This architecture resembles that of Dunn [Dun06] and Rerum [SWC+15]: the

redirection is part of all proposals, although an enforcement point and XACML are not directly mentioned in [WJ12]. A benefit-risk calculation and formalization is missing in [Dun06,SWC+15] but found in [WJ12]. The proposals in [Dun06,WJ12] do not consider a real-time data consumption stoppage by revoking the consent, which is formulated in [SWC+15]. Only [WJ12] was implemented and evaluated over an increasing scale of patients and requests.

In summary, consent management is a maturing technology. Advances can be found in health care scenarios, especially in the presentation of informed consent and benefit-risk evaluation. Architectural extensions for IoT scenarios are marginal, but the overall architecture has been similar through all proposals and can be applied to IoT as well. How well existing consent managers can be carried over to IoT, particularly those of health care, remains an open question.

4.2.2.2 Integration in the Rerum Domain Model

The architectural location of the consent manager is discussed here and subsequently put into relation with the consent manager via the sequence diagram shown in Figure 4.5.

In the Rerum Middleware the service manager is located conceptually at the Security Center, he interacts with the Rerum Devices and the data processing parties above the Middleware, see Section 2.5.4. The consent manager is essentially a part of the Security Center as it interleaves with other security services such as the authorization framework. For simplicity, it is assumed that the consent manager can create authorization policies and access information (such as tokens[9]), and that the VRDs can evaluate this kind of information. Figure 4.5 depicts the sequence of consent gathering based on a formerly unknown and unauthenticated request.

[9]Tokens, tickets and similar artefacts are structured access control information of authentication, authorization and accounting services. This topic is addressed in detail in Section 4.2.4.

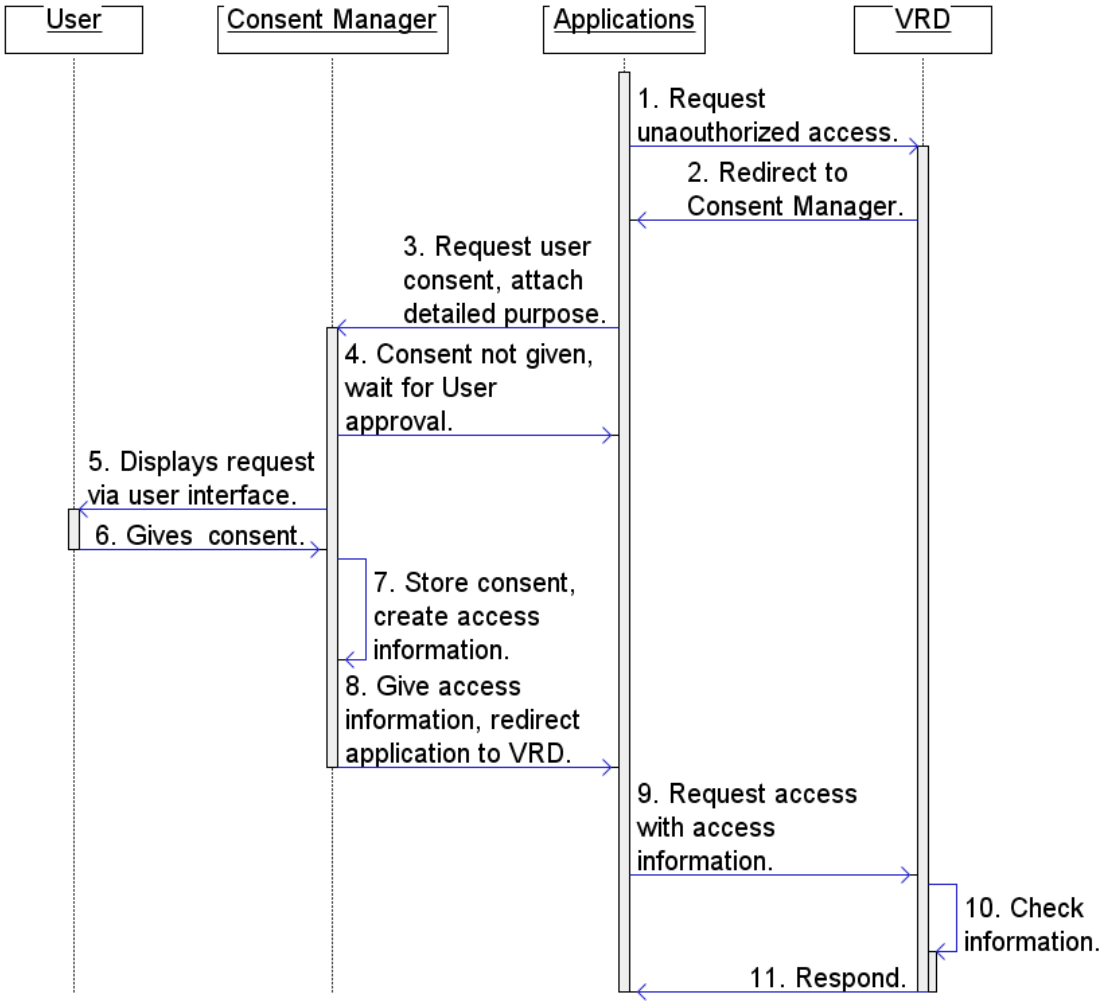

Figure 4.5: Sequence of the acquisition of user consent

Assuming that a new application requests access to personal data from a VRD, the functional interplay between the components is as follows:

1. A new application requests access to a set of private data residing in the Virtual Rerum Device. The VRD checks for access policies. At this point, there are none, as the service has never been authorized before.
2. The VRD redirects the request to the Consent Manager, as the user has to decide on granting access. Possibly, a user-defined policy could be used to give consent to certain applications. For instance, the policy could state that access is granted only to certain statistics of the user's data or only to requesters with reputation ranking of high or above and with a certificate from a certain given trusted group.
3. The Consent Manager receives the access request and asks the application for

the specific purpose for requesting the data, how the data will be processed and which data it wants to gather specifically.

4. The Consent Manager checks the request for policies or existing consent. The application is notified that none was found and that the user has to approve the request.
5. Upon checking the request, the Consent Manager includes the purpose information in a notification message and sends it to the user. The Consent Manager will wait for the approval of the user.
6. The user gives his consent. He accepts this service to access specific data for a certain purpose. The consent is recorded. (Please notice that besides this consent, an access control layer is also on effect. It is reasonable that in some cases, the accessing service must present particular credentials of the user on behalf of which he is accessing the data in order to grant some information to that user. Those credentials should be evaluated at the access control policy enforcement point).
7. After successfully gaining consent from the user, the Consent Manager triggers the creation of access policies for this application, including the data it is allowed to access, the way the data should be processed and the purpose.
8. The Consent Manager then redirects the application to the VRD and
9. Requests access with the attached access information.
10. The VRD checks the given access information.
11. On success, the VRD allows access.

Figure 4.6 shows a snapshot of Rerum's consent manager in the context of the indoor use case in Tarragona.

Figure 4.6: Rerum Consent Manager: Example for consent request in the UC-I2 trial

In Rerum, the consent manager interleaves with the access control layer to trigger the creation of policies, to authenticate the user, retrieve his consent, to validate if a request shall be automatically consented by means of user policies, etc. Differences and similarities with the access control layer are further addressed in [SWC+15].

4.2.3 Privacy Friendly Access Control

This Section describes how access control can be realized in IoT[10]. The need for access control was already mentioned in relation to the consent manager. User consent is an agreement between a user and a processing party on a purpose, which describes why (purpose) personal data is collected and processed. This purpose needs to hold at all times, whenever personal data is processed. The requirement of "purpose" is realized by the definition of Privacy Policies as formulated in the XACML 3.0 privacy profile [Ris13].

These policies can be enforced before disclosing data by checking if the requesting party can fulfil them. The enforcing component is called Policy Enforcement

[10]Note: the content of this Section has been previously published in [SC15, RER14b].

Point (PEP). There are two main cases, where an enforcement point can be placed.

- Directly at the Devices, the Privacy Policy Enforcement Point (PEP) will check fulfilment of an adequate policy for a certain requested data set.
- Decoupled, a trusted Privacy PEP will check policy protected data in transit. The trusted Privacy PEP has to know the adequate policy for this data set or the data set carries it alongside or has a link to the adequate policy, see Section 4.2.3.1.

In Rerum both privacy PEP versions were considered. The first one is integrated in the Virtual Rerum Device, as depicted in the snapshot of the Rerum middleware in Figure 4.7. The second one can be deployed in Rerum's Security Center, universally available for checking the data in transit.

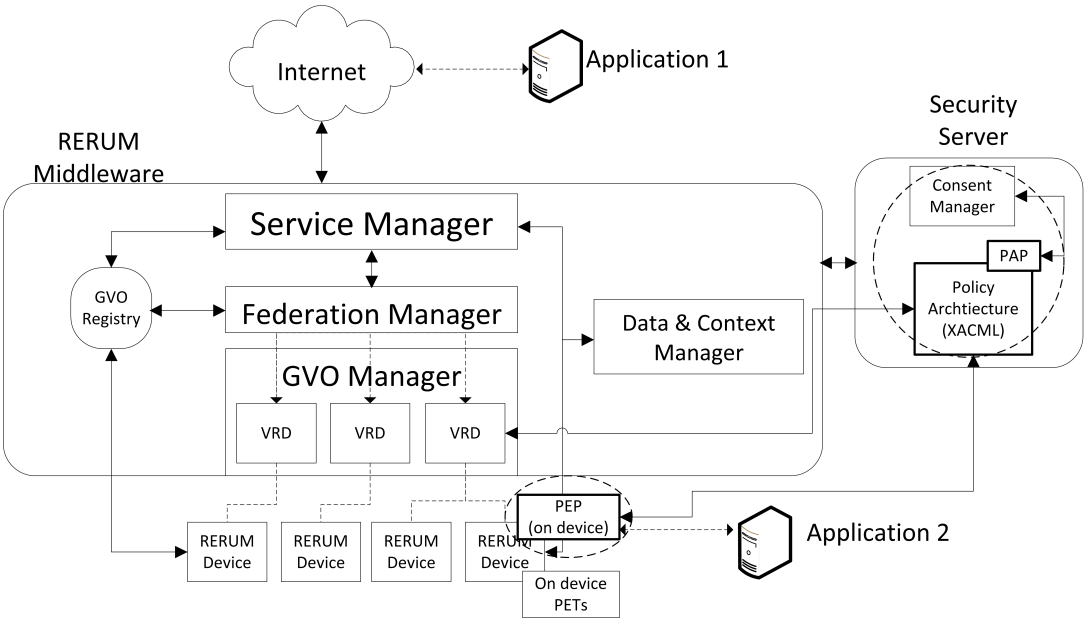

Figure 4.7: The access control modules (outlined) of the Rerum architecture

Figure 4.7 showcases the three different questions that have to be taken care of for access control to takes place in IoT. Rerum follows the XACML [Ris13] scheme. Anticipatory, three main components will be mentioned here: the Policy Enforcement Point (PEP) where access control is enforced, the Policy Decision Point (PDP) where the decision to either allow or deny a request is made and the Policy Administration Point (PAP) where the policy or information is defined that will be used to make a decision.

The first question concerns the integration of the XACML scheme into the Rerum domain model. This is briefly explained in the next Section alongside with an introduction to XACML. A full picture of the domain model with XACML is presented in Chapter 5. The interested reader is referred to [RER15] in order to grasp the full functional integration with the Rerum middleware.

The second question relates to the enforcement point: in IoT an application can either request data from a device that maybe physically nearby or it may refer to the middle ware to check which (physically distant) device can provide certain information (compare the applications in Figure 4.7.). Therefore two PEPs are needed as mentioned in the introduction of this Section. The PEP on the middleware follows the XACML standard, although it has to rely on additional information for data in transit. PEPs on the devices however, are constrained in computing power and network connection. A device has to either make a decision by itself and employ lots of processing power or delegate the decision to a party that it trusts and rely on ubiquitous network access. A novel way of finding a trade-off between both has been developed for this thesis and proposed for Rerum, see Section 4.2.5.

The third question in focus for the middleware PEP is how to link policies to data in transit. Policies can either be known by the PEP (and the PDP), linked or stuck to the data. The scheme developed for Rerum is discussed in Section 4.2.4.

In the next Section an introduction to related work and basic understanding of used mechanisms will be given.

4.2.3.1 Related Work

In this Section related standards that were used in Rerum will be presented, namely policies for data in transit, the Extensible Mark-up Language XACML, the access control protocol Open Authorization standard (OAuth) and the Delegated CoAP Authentication and Authorization Framework (DCAF). For completeness, the related standards Security Assertion Markup Language (SAML) and Kerberos will be briefly discussed.

Sticky Policies. The sticky policy mechanism suggested in [PM11] will be described here, which aims to allow access to personal data only upon satisfaction of the attached policies. These is achieved by encrypting the data set and disclosing decryption information to parties fulfilling the policies. The sticky policy mechanism can be described by three basic steps, as shown in Figure 4.8.

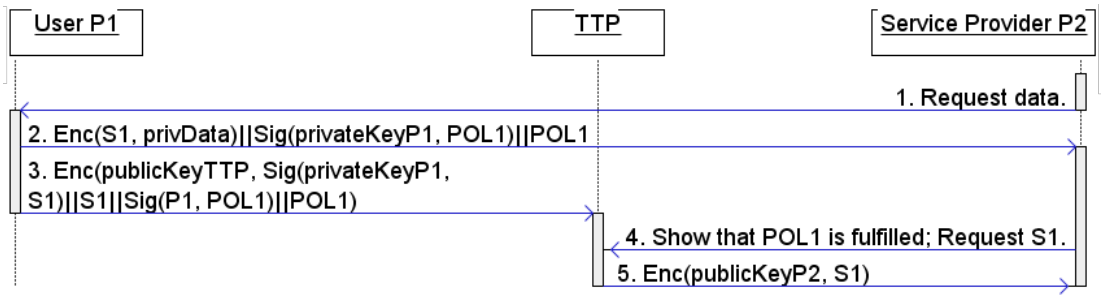

Figure 4.8: A simple Sticky Policy mechanism

Three parties are assumed: person one ("P1") is the data subject creating data sets, the second person ("P2") is the data controller processing the data, and the third person is a Trusted Third Party (TTP), which is able to verify that the data controller fulfils policy obligations.

Step 1 P2 requests personal data from P1. P1 generates a data set *privData* and according policies *POL1*. The data set is encrypted with a secret *S1* and the policies are attached as metadata to the encrypted data. Alternatively, the policies could be stored in a public registry with only a policy pointer sticked to the data set's as metadata. Person one signs the policy with his private key *privateKeyP1* and sends the data, the policy and the signature to P2.

Step 2 P1 sends an encrypted message to the Trusted Third Party with S1, his signature over S1, POL1 and its signature.

Step 3 P2 wants to access the data set, which is encrypted with S1. P2 understands the attached policies POL1, he requests S1 from TTP, showing that he can fulfil the requirements from POL1. P2 receives S1, if TTP is convinced that P2 can fulfil the policies satisfyingly.

It should be noted, that in this small example, there is no need for a Trusted Third Party, P2 could ask P1 himself for S1. In case of data in transit through

multiple parties, P1 might not be available, thus TTP is assumed a party with much higher availability and connectivity than the data subject himself.

XACML. The Extensible Mark-up Language (XACML) [Ris13] is a standard XML scheme for authorization ratified by the OASIS group firstly in 2005. The standard focuses on the definition of authorization policies although an architecture proposal exists. At the time of this writing, the most recent version is profile version 3.0. A series of profiles have been proposed that help to apply the XACML scheme with other authentication schemes such as the Security Assertion Markup Language SAML [HCH+05] and the OAuth protocol [Sir14] in order to facilitate a complete access control framework.

The XACML scheme is defined through four main components. For understandability, a visual image is used to exemplify the scheme (Figure 4.9), a deep technical description is given in [Ris13].

Figure 4.9: Exemplification of XACML by mapping a real life example

Policy Enforcement Point. The PEP can be envisioned as a door that has to be opened for a requester to access a resource. The PEP is an dependent

component that cannot decide by itself if a request is to be allowed or not. But the architectural location of the PEP is independent, it can be scattered throughout a system wherever a request has to be validated. It should be noted that it is reasonable to place the PEP and the policies as close as possible to the resource. This is visualized in Figure 4.9, the door is ultimately placed in front of the property including its policy (sign).

Policy Decision Point. In the imaginative context of the PEP being a door, the Policy Decision Point (PDP) acts as the bouncer. The PDP decides upon the information it has been given if the request is allowed or not. Mandatory information for the PDP is: the resource that is requested (e.g. via a resource identifier), a policy that was defined for that resource by the owner (also sometime the possessor, see Section 2.5.3) of the resource and proof on who the requester is or whom he relates to.

Policy Information Point. The PDP might need further information to be able to successfully match a given policy to a request. The PDP is hereby assisted by the PIP. Additional information or tasks might be: resolving IP-addresses to geo-locations, retrieve further credentials to identify the requester, get time zones, get IoT context information, etc.

Policy Administration Point. The administration point is where policies are defined. The PAP is uniquely accessible to the owner of the resource. The PAP also serves as the policy store. In the image, it corresponds to a protected place where the owner of the property create the policy sign.

All the elements of the XACML scheme can be found in Rerum, an overview is given in Figure 4.10.

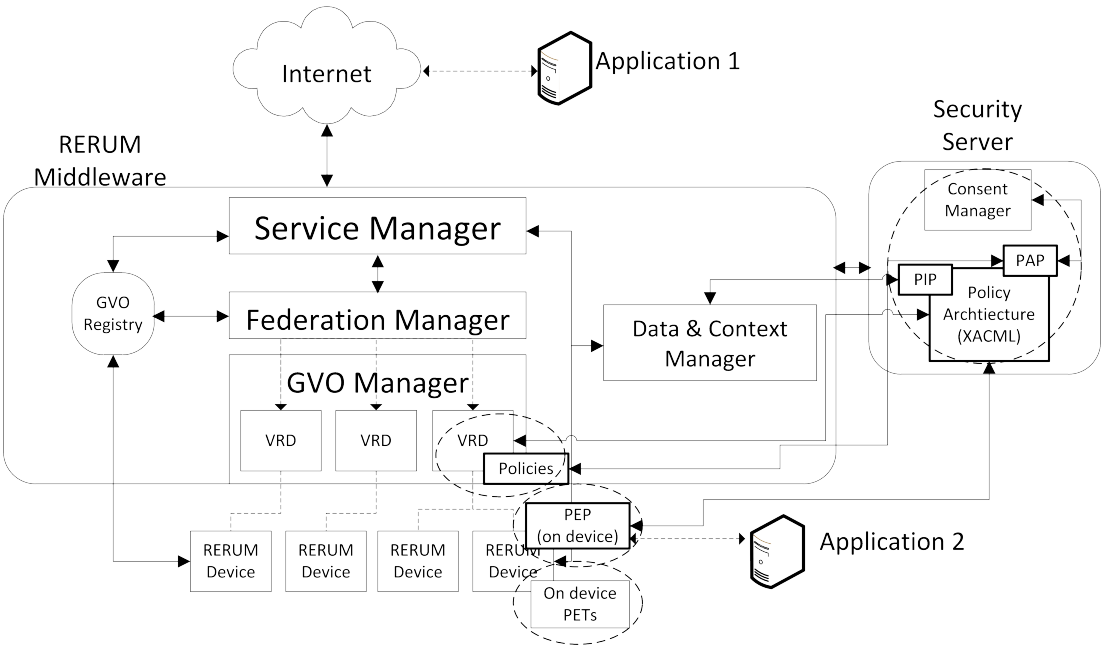

Figure 4.10: Integration of XACML components in the Rerum middleware

The XACML scheme is initiated in Rerum as follows:

1. An authenticated application requests access to data of a device.
2. The device does not decide itself upon granting or denying access, it redirects the request to the Policy Decision Point, located at the associated Virtual Entity.
3. The PDP checks which policies apply for the data requested and if the requester can fulfil them.
4. If needed, the PDP will request more contextual information from the PIP for the decision.
5. The PDP sends the decision to the PEP, which will act accordingly. There are three decision types that the PDP can send to the PEP: allow access, deny access, request undecidable.
6. The PEP will act according to the decision and either send the requested data or deny the access. (It is assumed "deny" as default, if no policy applies positively or in case of conflict.)

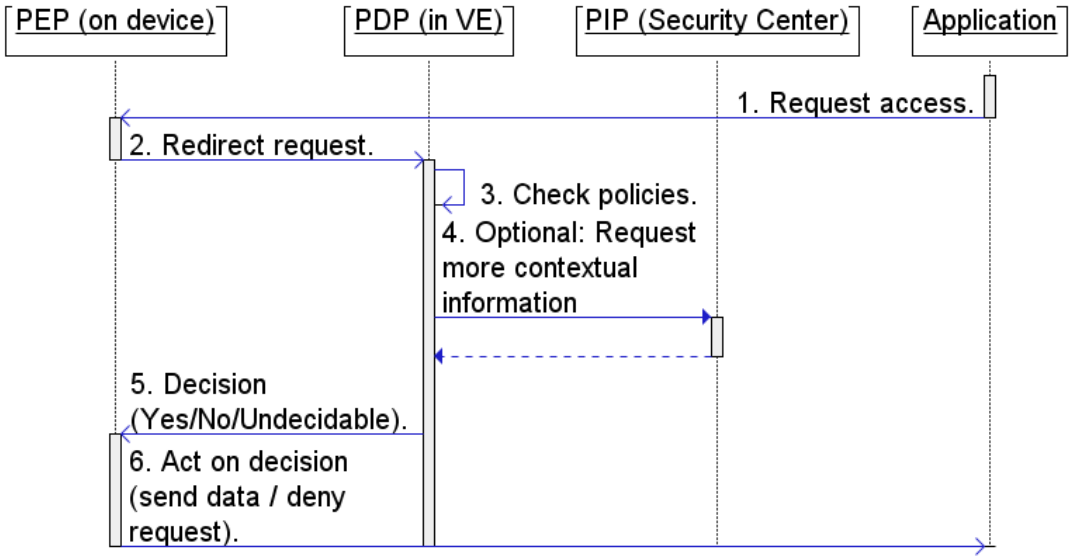

Figure 4.11: Sequence for the access of protected data in Rerum through XACML

Figure 4.11 diagrams the sequence. It should be noted, that while the policy enforcement point and the policy definitions for privacy and security are functionally different, the underlying infrastructure is the same in Rerum.

OAuth. The OAuth framework (RFC 6749) [Har12] is an open standard for authorization ratified by the IETF firstly in 2010. At the time of this writing, the most recent version is version 2.0. OAuth and XACML standardize different aspects of authorization, while XACML specifies policies which are used to reach a decision concerning access requests, OAuth standardizes the sequence of messages transferred between the parties involved. OAuth provides a protocol for requesters (called clients) to access so called resource servers on behalf of resource owners, such as: a smart phone (client) accessing a resource (heart rate sensor) on behalf of a user in an ownership relationship (user owns client and resource).

It also provides a process for users to authorize third-party access to their resources without sharing their credentials. Notably, OAuth leaves open what content exactly is transferred and how each party handles the OAuth messages to reach authorization decisions. For the message content, OpenID Connect [SBJ+14] has been proposed as an authentication layer on-top of OAuth. For decision handling, an interleave between XACML and OAuth has been proposed, see for example [Tha14].

In the following, the OAuth 2.0 protocol for requesting and renewing access

information (in form of tokens) will be described here on its main scenario of *delegated access*. The involved parties (or Roles, as OAuth defines them) are described first before outlining the OAuth protocol itself. The Roles are:

Resource Owner (RO) The owner of the resource which will be accessed. The owner is capable of giving access either due to the ownership relationship of it and the resource (then it is called a user), see Chapter 2.5.3, or due to a delegated ownership form the user (or organization) to some device that will be representing the user as the owner of the resource.

Resource Server (RS) The server hosts the protected resource. The term "server" suggests a powerful entity with high connectivity and computing power, but that is not necessarily the case. The resource server can also be a small IoT device that hosts the information it reads as the protected resource. The server is capable of understanding access control information and responding to requests that target protected resources.

Client (C) An entity that makes requests on behalf of the resource owner and with the owner's permission. The client is an unspecified entity that could reside on another resource server, a desktop or any other form of software artefact.

Authentication Server (AS) The server issues the access control information in from of OAuth tokens to clients. The client needs to authenticate as a representative of the resource owner against the authorization server. Additionally, the resource owner must have authorized the client previously for the client to receive a token. The authorization server is an conceptual entity: it may be the same server as the resource server or a separate entity. Also, a single authorization server may issue tokens accepted by several resource servers.

OAuth messages flow between the roles client, resource server and resource owner as summarized in Figure 4.12.

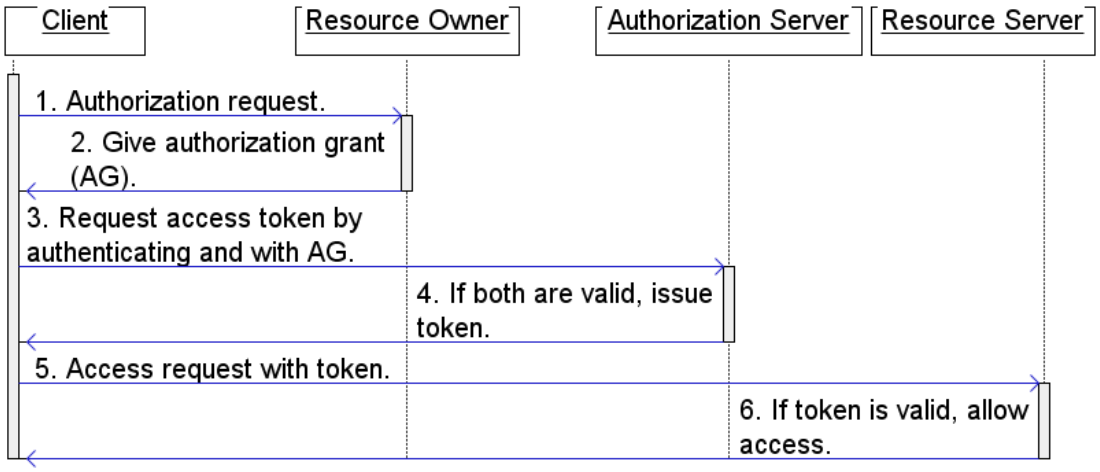

Figure 4.12: Sequence of the OAuth 2.0 protocol

The protocol starts with the client. The client knows the resource owner (message 1) and asks it for the authorization to access its resource. The resource owner decides (how the decision is made, is out of scope) and gives Access Grant (AG) information to the client. The content of the grant is out of scope as well. The client requests an access token from the authorization server by authenticating itself and attaching AG (message 3). If both are valid, the authorization server issues the access token (message 4). The authorization type, expiration date and other constraints refer to the information of the access grant. Message 5 and 6 represent the authorized access request by the client and the successful response by the resource server.

The OAuth standard notes explicitly that the interaction between the authorization server and resource server is beyond the scope of the specification [Har12], as is the specific content of the token. This interaction is not trivial, as both servers have to agree on a common understanding for a variety of clients, resources, contexts and conditions. The missing interaction between both servers inspired the proposal in Section 4.2.5 for constrained IoT devices.

DCAF. The Delegated CoAP Authentication and Authorization Framework (DCAF) [GBB14] was firstly drafted for the IETF in 2013. DCAF specifies a protocol for establishing a Datagram Transport Layer Security (DTLS) channel in constrained environments. DCAF uses the same roles as OAuth, but specifies the delegation of client authentication and authorization from a constrained client to

a more powerful role called Authentication Manager (AM)[11]. DCAF additionally focuses on the use of CoAP, the Constrained Application Protocol, which was introduced in Section 2.6. The authentication manager is again a conceptual role, as AM and client can be one role in DCAF as well to fully resemble the OAuth protocol. The protocol flow is as shown in Figure 4.13.

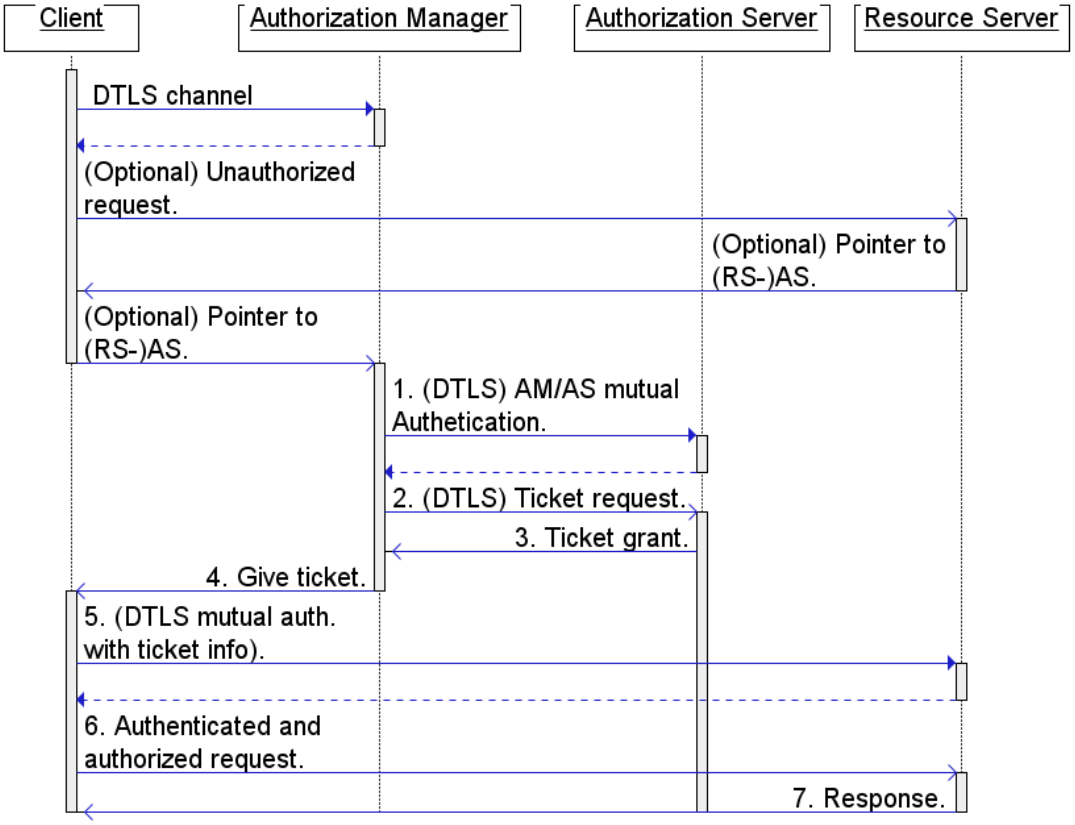

Figure 4.13: Sequence of the DCAF protocol

Notably, the resource owner does not appear in the sequence. DCAF maintains the role of the resource owner as the principal entity that defines access permissions to the resource server, but does not focus on how it does so. Therefore, the role of the RO is taken out of the protocol specification.

The DCAF protocol may start in two different ways, by either the assumption that the client knows the RS' related AS or by the client requesting the related authorization server of the resource server. Figure 4.13 starts the protocol with the latter.

The client sends an unauthorized (and unauthenticated) request to the Resource

[11] If interpreted carefully, the notion of delegation was already introduced in OAuth, as the authorization server and the resource owner take care of authentication and authorization for the resource server.

Server. The RS responds with a pointer to its Authorization Server. The client understands this response as the need to request more access information from the client's Authorization Manager. Here is where the protocol definition for requesting access information starts: the AM requests a DTLS connection to the AS (message 1). By means of public key infrastructure or otherwise (this is out of scope for the protocol), AM and AS mutually authenticate and establish a DTLS channel. Over the DTLS channel, the AM requests access information in form of a ticket (message 2). The resource owner has deposited some policy on the AS, such that it can decide if the AM is to allow the request. Assuming that it is, the AS issues a ticket to the AM (message 3). The AM transfers the ticket to the client (message 4). The client can now request the establishment of a DTLS channel with the RS. The key material is hereby provided to the client in the ticket. The RS has a priori knowledge of key material (message 5). The client requests access to the resource (message 6) which is responded due to a valid ticket (message 7).

DCAF partially defines some of the possible content of the ticket and interactions between the delegating parties, although leaving privacy aspects out of scope. Adding privacy and retaining efficiency for constrained devices is thus the main contribution of Section 4.2.5.

4.2.3.2 Other Authentication Protocols

For the sake of completeness the Kerberos protocol and the Elliptic Curve Qu-Vanstone Implicit Certificates are briefly mentioned.

The *Kerberos protocol* (RFC 4120) [SNS88] is an open protocol for authentication and authorization firstly proposed in 1978. At the time of this writing, the most recent version is version 5.0. Kerberos was designed to allow authentication over unprotected networks. The protocol was extended to add additional information regarding authorization for the involved parties. The protocol partly relies on the Needham and Schroeder's Trusted Third Party authentication protocol [NS78]. Kerberos uses a strong attacker model, which is able to break assertions by a host's operating system, modify host addresses, break physical security of all the

hosts on the network, and read, modify and inject network packets at will. The protocol does this in the following (simplified) way:

Kerberos is purely based on symmetric cryptography. There is no definition of a key distribution, therefore the PKINIT Kerberos extension [ZT06] (RFC 6112) may be used. A scenario with three parties is assumed: a Client, a Server and a Trusted Third Party (AS). The AS shares a symmetric key with the Client ($K_{AS,C}$) and the Server ($K_{AS,S}$) respectively. The Client and the Server share no secrets, although this is the goal of the protocol. The Client asks the AS for a shared secret with the Server. The Client does this by encrypting its request with the the common key ($K_{AS,C}$). If decrypted successfully, the AS responds with a shared key for the Client and the Server ($K_{S,C}$), encrypted with the common key of Server and AS ($K_{AS,S}$) and the common key of Client and AS ($K_{AS,C}$). Consecutively, the Client can decrypt the new common key of Client and Server generated by the AS, but it cannot modify the message sent to the Server[12].

Additionally, the AS can also send an attachment in the encrypted block that the Client receives and that is intended for the Server. This attachment may contain authorization and identification information such as the name, IP-address, access permissions, etc.

The protocol ends after the Client has sent the encrypted message of AS to the Server. The Server successfully decrypts the message and sends an acknowledgement with the new symmetric secret ($K_{S,C}$) to the Client.

The *Elliptic Curve Qu-Vanstone Implicit Certificates* [Res14] is an asymmetric certificate scheme, based on Elliptic Curve Cryptography [HMV06] and has been firstly proposed in 2004 as an intellectual property item (patent) by Qu and Vanstone [QV04]. The scheme involves a Certificate Authority (CA), a claiming party "A" and a verifying party "B". It works as follows: A proves its identity to the CA. A receives some cryptographic material from CA, which, together with A's identity, will serve as A's private key (as in a private-public-key scheme). A wants to prove its identity to B. A creates a signature with the cryptographic material previously received form the CA and the information "$IamA$". Additionally, With

[12]Here, "Kerberos encryption" also assumes integrity protection. This is not necessarily the case, as RFC 4120 explicitly refers to this problem and different modes of operation.

the information of the proclaimed identity "A" and CA's private certificate, B can construct A's public key and verify the signature and, implicitly[13], A's identity. This is the goal of the scheme.

Kerberos and the Implicit Certificate scheme have several interesting properties for IoT. Kerberos is a well understood protocol where the protocol design reference implementations have been studied for over 30 years. The protocol is purely based on computation efficient symmetric cryptography, which fits to the constraints of IoT devices. Still, Kerberos data flows have to be adapted for IoT, as each commonly generated secret and access information (so called *Ticket*) is only usable for one single partner. Implicit Certificates are based on Elliptic Curve Cryptography which generate small sized certificates and also efficient computation of verification and sign operations, although those certificates are still considerably larger than the symmetric material Kerberos and other protocols use. A more complete overview and comparison of these protocols is therefore presented in [Che15] and summarized in Table 4.2.

4.2.4 Sticky Policies for Data in Transit

In this Section policies for data in transit are integrated into the Rerum domain model. The content of this Section has previously been published in [SC15, SWC+15].

In order to support the privacy principles of Purpose Legitimacy & Specification, and Individual Participation, see Chapter 3, the user should have a way of expressing his privacy choices to the system, in the form of user-defined privacy policies. It is important that the policies are easy to retrieve when needed. This is not so easy to guarantee in an IoT-based system with huge amounts of data. For this purpose the policies should be linked to physical entities they relate to, when the data is at rest. Data in transit should travel with sticky policies, as introduced in Section 4.2.3.1.

[13]The aspect of implicity comes with the idea that B needs A's identity to successfully construct A's public key.

4.2.4.1 Integration in the Rerum Domain Model

For the integration of policies, the reader is breifly reminded how IoT-A and Rerum relate Physical Entities to Virtual Entities.

The Rerum domain model refers to physical objects as Physical Entities. PEs can be any object in the "real world" including living things. PEs are represented as Virtual Entities in the virtual space. The relationship between Physical and Virtual Entities is defined as a conceptual entity called Augmented Entity. If an AE exists, the corresponding Virtual Entity changes according to the Physical Entity. If the Physical Entity changes physically, the values of the Virtual Entity are updated. Virtual Entities expose resources that can be requested by clients, for example, by service providers. Note: the Virtual Entity could also change the Physical Entity, if actuators can react accordingly.

A natural way of binding the privacy policies to a Physical Entity is to link them to the entity in its virtual representation. The policies can now be enforced on access requests for a Physical Entity, as the requests have to be directed to the virtual representation. Sticky Policies are privacy policies that are attached to data and accompany it whether it is stored or in transit. This policies promote the user's wishes of allowed actions and consent obligations for parties processing the data.

In the Rerum domain model, sticky polices can be located on the Virtual Entities as well. The reason is that the Virtual Entities are the earliest point of data creation where policies can be stuck to the data as soon as it is created.

The integration of sticky policies in the Rerum architecture relies on the policy generation as described in Section 4.2.3.1 (XACML). Figure 4.14 illustrates where data is protected and policies attached.

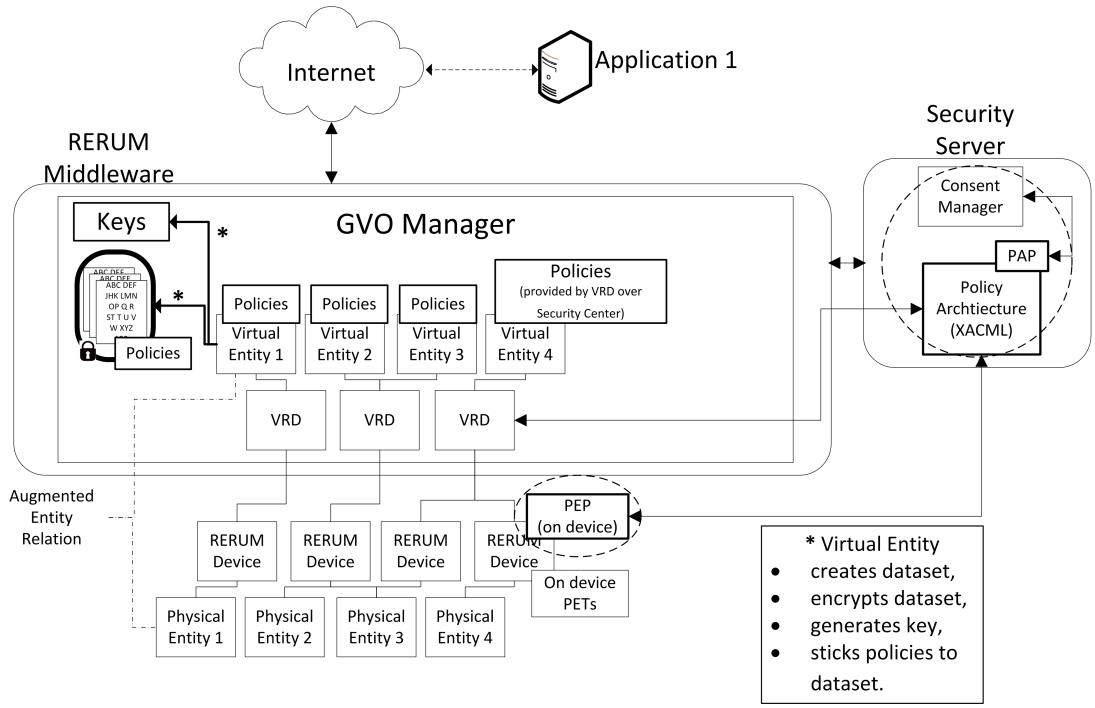

Figure 4.14: Sticky Policies in the Rerum domain model

For the application of sticky policies, policy generation and the provision of data is needed. Datastreams, which are not protected by sticky policies, are provided by Rerum devices, while pre-processed datasets are provided at the Virtual Entity. Policies are stored per Physical Entity at the corresponding Virtual Entity (see Rerum Deliverable D2.3 [RER14b], Section 6.11.2.2). Therefore, a protected dataset can be generated at the Virtual Entity. That means a dataset is encrypted, policies are attached and then both are sent to a requesting party. The corresponding secret is sent to either a Trusted Third Party, which could be another more powerful device of the data subject, or to a global privacy enforcement point at the Rerum Security Center. Exercising the generation of a sticky policy protected dataset, the Virtual Entity would follow these steps:

Step 1 The Virtual Entity is requested a multiparty dataset. Multiparty datasets are always protected by sticky policies. The Virtual Entity generates the dataset and a corresponding secret S1. The policies to be attached are taken from the policy database of the Virtual Entity.

Step 2 The Virtual Entity encrypts the dataset with S1 and attaches the policies to the dataset. The Virtual Entity signs the policies with its private key (or with another secret which is verifiable by a public counterpart).

Step 3 The encrypted dataset, the policies and the signature are sent to the requesting party.

Step 4 The secret S1 and the policies are again signed by the Virtual Entity and sent in a confidential way to the Trusted Third Party. The TTP in Rerum could be a device of the data subject which has a higher availability and connectivity or a trusted service found in Rerum's security and privacy center.

Step 5 The requesting party shows to the Trusted Third Party that it can fulfil the requirements of the sticky policy. The TTP provides the secret in a confidential way to the requester.

Depending on the policies, there might be many requirements to be fulfilled before acquiring the set's secret. Pearson et al. [PM11] describe following possible policy requirements:

1. Specify proposed use of the data. Example: use only for research and transaction processing.
2. Use of the data only within a given set of platforms with certain security characteristics, a given network or a subset of the enterprise.
3. Specific obligations and prohibitions such as allowed third parties, people or processes.
4. Blacklists, notification of disclosure and deletion or minimization of data after a certain time.
5. A list of Trusted Authorities (TA) that will provide assurance and accountability in the process of granting access to the protected data, potentially the result of a negotiation process.

It should be noted that sticky policies first and foremost describe the obligations needed to process the data, but they cannot prevent misbehaviour after the data has been decrypted.

Sticky policies are a soft mechanism for privacy protection that allows service providers to be compliant with the user's consent and to respect a user's wish for privacy. Sticky policies are used to attach policies to data, protect a data set until a service provider proves that it fulfils privacy requirement (this works up to a

certain point), and allow a service provider to respect a user's wish, even with data sets from an unknown user.

4.2.5 Privacy Enhancing Tokens

This Section briefly details the IETF draft *Privacy-Enhanced Tokens for Authorization* firstly proposed in 2015 the Authentication and Authorization for Constrained Environments IETF Working Group (ACE)[14] [CSH15]. Additionally, new results on energy efficiency and the formal verification of the protocol that were made during and after the first and second versions of the draft[15] are summarized. A part of the content of this Section has been published for Rerum in [SWC+15].The draft is lead and maintained in the IETF by Siemens AG.

The IETF has classified devices by their computation and storage capacities as introduced in Sections2.6.2 and 2.6.1. The most used devices in IoT are "class 1" with a working memory restriction of *10 KB RAM* and a storage capacity of *100 KB Flash*. Class 1 devices are typically powered by coin or dry cell batteries with a maximum capacity of *2376 joules*, see [Dev11].

The classes are not clear cut and other classes are used in some cases for IoT. For the elicitation of adequate privacy enhancing technologies, the constraints of class 1 are those that will impose a significant factor.

Smart devices in the Internet of Things range from desktop computers to small hardware embedded in everyday objects such as clothing. As previously discussed, not only are these small computers constrained in computing power and memory but also have economical limitations. Therefore it is difficult to integrate state-of-the-art privacy technologies. Thus, the following proposal targets authorization and exemplifies a currently researched approach that allows that mechanisms to be fulfilled.

4.2.5.1 Efficient Privacy Friendly Authorization

Many services will consume data from IoT devices and, accordingly, the authorization of those services against the devices will be needed. Authorization frameworks

[14]Draft abbreviation: draft-cuellar-ace-pat-priv-enhanced-authz-tokens.
[15]This results have been published in [Che15] and on IoT Week 2016.

such as SAML and OAuth[16] are based on tickets/tokens, which require an authorization party to fully understand the token and deny or allow the request based on policies. A device with 10 KB RAM and a storage capacity of 100 KB would be very constrained in managing the tokens, policies and its functional processes for measurement, actuation etc. IoT devices therefore need a light-way approach for authentication.

The basic idea is to *delegate* the "heavy" computational effort, such as policy validation, access decision and token generation, to another component. An access request from a service provider to a device is delegated (by the device) to its "authorization manager". The authorization manager is an entity that knows the device's resources and policies and decides upon access or denial of the request. If the decision is positive, a token/ticket is generated and sent back to the service provider. The token/ticket information is generated in such a way that the device can unequivocally identify an access permission for a respective request. The service provider sends a new request with the token/ticket to the device. If the request matches the ticket/token information, the request is granted.

This approach is depicted in DCAF, see Section 4.2.3.1. The DCAF architecture defines a constrained device as a "resource server", if the device senses data and "serves" it to service providers and clients. Accordingly a Server Authentication and Authorization Manager (SAM) is defined, which is the delegation component assigned to the server.

A client or service provider has to obtain a ticket or token from the SAM[17] first to request data from the server. The server has a white-list approach: if the ticket or token is known to the server and the request matches the token, the access is granted, else, the access is denied.

However, building token material that does not identify a client every time it wants to consume data from a server is a further challenge. This privacy enhancement aims to protect the client from being constantly identified when it is using the same (or an identifiable) token.

As delegation takes place, privacy mechanisms have to be integrated in both

[16] OAuth itself only targets the protocol flow, it is OpenID Connect that puts an identification, authentication and authorization layer on top of OAuth, see [SBJ+14].

[17] Ticket or token depends on the underlying authorization protocol.

components, Server and SAM. An extension with privacy friendly token material is therefore proposed[18]. Here, the authentication between SAM and client is done in a privacy enhanced way, e.g. with group signatures. The tokens are privacy enhanced by constantly changing secrets. The secrets are generated in such a way that the constrained server is still able to identify them by itself with only little resources, even if they are often changing. Both approaches, DCAF and privacy enhanced tokens as an extension, are actively being developed in the IETF ACE working group. They show how delegation can help to overcome computational constraints and how it might be a building technology for further privacy enhancing technologies in IoT.

Protocol Overview. The Privacy Enhanced Tokens or Pseudonym-based Authorization Tokens (PAT) protocol extends DCAF. The reader is referred to Section 4.2.3.1 where the DCAF protocol was outlined.

The protocol uses the same roles as DCAF. For simplicity, the resource server is simply called "Server" or abbreviated Server (S), the authorization manager which interacts with the resource server is called "(Resource) Server Authentication and Authorization Manager" or abbreviated "SAM" and the authentication manager for the Client is called Client Authentication Manager (CAM) as well. Arguably, SAM and CAM support the Server and the Client with authorization and authentication, therefore the "A" stands for authentication and authorization. The PAT protocol adds token information related to each role:

- Server Token (ST): the token which is generated by the SAM for the Server. Besides parameters and which may contain authorization information that represents the Resource Owner's authorization policies for C, also contains a secret, St, called the $ST-secret$. This secret can be used to verify the Authorization Token and to generate other secrets which are discussed later.
- Client Token (CT): the token which is generated by the SAM for the Client. It contains a secret, Ct, which can be used to generate the Authorization Token. Optionally CT may contain authorization information that represents RO's authorization policies for C.

[18]This proposal has been fristly published in [CSP15].

- Authorization Token (AT): the token which is generated by the Client and presented by him to the Server. It contains a secret AT, which changes regularly (in a similar way to one-time passwords). The AT contains all information needed by the Server to verify that it was granted by SAM.
- VerifK, PSK, IntK, ConfK: derived keys between C and S used respectively to (i) verify that they are talking with the intended partner, for the Client C it is used as proof of possession of the (current) Authorization Token, (ii) as pre-shared key to establish a DTLS secure channel, (iii) for integrity protection in message authentication codes and (iv) for confidentiality protection (to be elaborated in a future version of the draft).

S and SAM and C and CAM are assumed to have a secure channel preserving integrity and confidentiality.

Using this secure communication channel SAM provides to S a main secret x which is used within the initial version of the Server Token (ST). The server token $ST = St, paramS$, where St is a secret created by SAM (in a way that is outside of the scope of the draft), and $paramS$ is a set of parameters, determining the functions G, $g1$, $g2$, $g3$, $g4$ that are discussed later and, optionally, the authorization policies for the clients. To gain access to a specific resource on a Server S, a Client C requests a token from the SAM, either directly or using its CAM. In the following, for simplicity, only the collocated CAM-C role is discussed; the separation of the roles was explained in Section 4.2.3.1 to the reader.

After SAM receives the request from C, he decides if C is allowed to access the resource. If so, it generates a Client-ID and a corresponding Client-Token used for the authorization and for securing the communication between C and S.

For explicit access control, SAM adds the detailed access permissions to the token in a way that C or his CAM can interpret and S can verify as authentically stemming from SAM.

Then C presents the Authorization Token to S, demonstrating his authorization, and C and S can establish a secure channel.

Message Flow Overview. In Figure 4.15, a PAT protocol flow is depicted. A noteworthy difference in comparison to DCAF is the different order of messages 7 and 8 and that the DTLS channel between C and S for messages 10 and 11 are optional. Note that other native PAT methods (g* functions) could be used besides DTLS to do this.

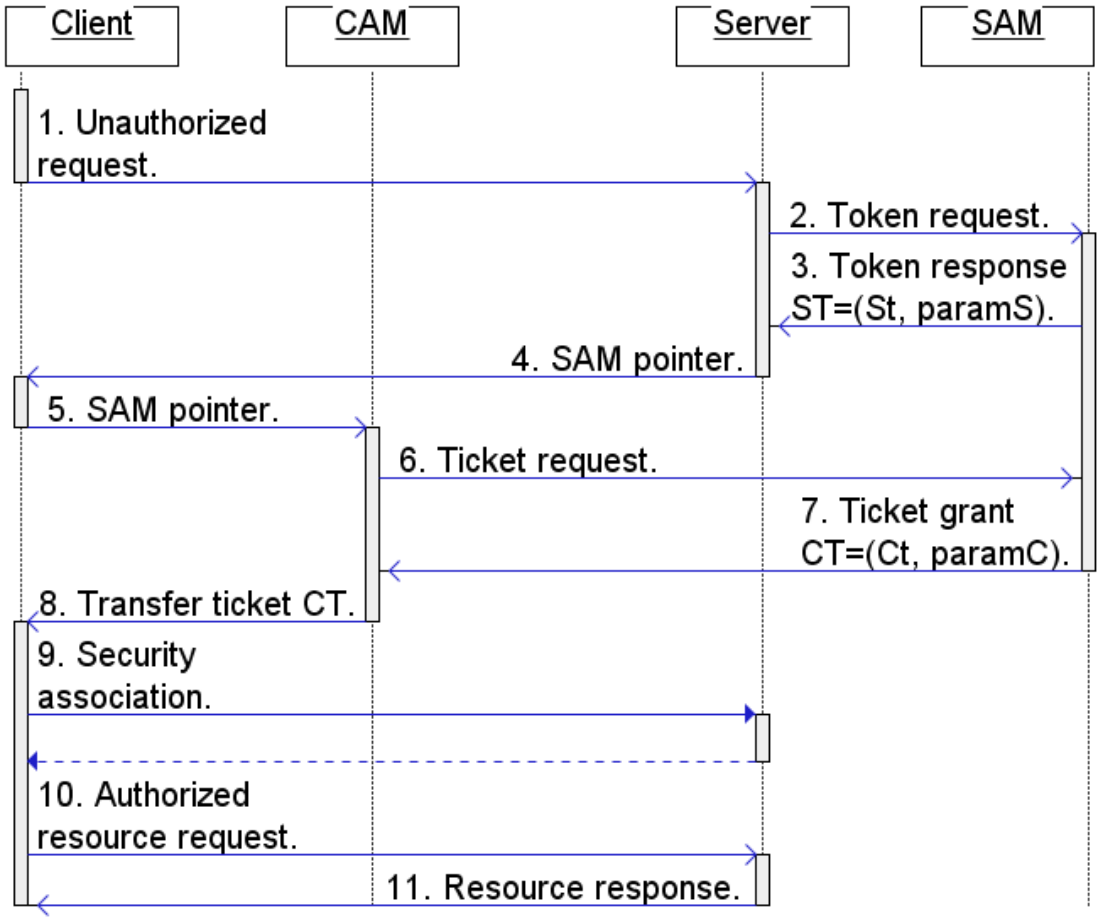

Figure 4.15: Pseudonym-based authorization tokens

It is assumed that the Server S and its Authentication Manager SAM share a secure channel. This is not necessarily a (D)TLS channel, it could also be a physical or near field communication, e.g. by transmitting the information via an USB stick or by putting one device physically (very) close to the other. No particular concrete secure channel is therefore described, but it must be stressed that *the security of the protocol strongly depends on how the security of this channel is designed and implemented for the communication between SAM and S.*

It is also assumed that CAM and SAM share a secure connection, say over DTLS.

As in DCAF, to determine the SAM in charge of a resource hosted at the

S, C may send an initial unauthorized request (message 1) to S. This optional unauthorized Resource Request message is a request for a resource hosted by S for which no proper authorization is granted. S must treat any request as an Unauthorized Resource Request message when any of two following holds:

- S has no valid access token for the C and the requested resource.
- S has a valid access token for the C, but the token does not allow the requested action on the requested resource.

S then denies the request and sends the address of its SAM back to C (message 4, also possibly by asking the SAM itself for the pointer, messages 2 and 3). This message is also found in the original DCAF protocol. Or, instead of the initial unauthorized request message, C may look up the desired resource in a resource directory that lists the available resources (this option is not shown in the sequence diagram).

Message 3 is especially important: while retrieving the pointer information for C, the server could also provide what is considered the security anchor of this protocol. The owner of the server determines a number N which is (probably) an upper bound on the number of Clients that the Server will simultaneously serve. This number N should not be too high, as the storage and computation effort of the server[19] will increase (linearly) with N. The protocol does not restrict changing N in any state, N can be increased or decreased if necessary. Using the secure channel, SAM sends to S the initial value of ST = (St, paramS), where St is a (preferably, random) number that can't be guessed by an attacker and paramS is a set of parameters that encode the number N, the choice of functions G, g1, g2, g3, g4 and the permissions Client Nr "i" has been assigned (either for each Client i or for a set of them). The permissions may remain undefined or incomplete and can be extended or modified later at any time. They may also contain validity periods or other restrictions in the Service Level Agreement. At any later point in time the SAM may change ST: send a new value for St, either change or extend the permissions or change the number of expected Clients N.

[19]The reader is reminded that the devices serving data in the area of IoT are mostly class 1 devices, see Section 2.6.

Once C knows SAM's address, it can send a request for authorization to SAM (directly, or as in messages 5 and 6 indirectly using its own CAM). The Client expresses hereby the set of permissions he needs to access the resources of the Server.

If the access is to be authorized, SAM generates a Client Token (CT) for C (or CAM as in message 7, which is then transferred to C in message 8). SAM decides which Client Number "i" the Client C should have. Each Client is assigned a different number. The number "i" is an integer between 1 and N, the number of Clients. The choice of value for "i" depends on which permissions the owner has foreseen and, more importantly, the SAM has encoded as parameters sent to S. The ticket contains keying material for generating all necessary tokens and keys. If necessary, a representation of the permissions C has for a resource is also added to the ticket.

With their common knowledge in St and Ct, C and S are able to establish a secure channel (message 9). This is were one of the functions g* is used.

Each time C sends S a resource request, it generates and presents a (current) Authorization Token to S to prove its right to access (message 10 and consequently message 11). In possession of the Client Token (CT) the Client can construct valid Authorization Tokens (AT) which demonstrates his authorization to access the resources he is requesting. Regularly, message 10 has to be sent afresh and a new AT must be used, that means, that C has to renew his authorization status at the Server. The frequency in which the Client has to send a new AT can be enforced by C and is determined indirectly by the owner of S (or by SAM). This allows a fine-grained control on the service level that the Server will provide to the Client (for instance, on the amount of information of sensor data). It is assumed that the frequency of renewal is the same for all Clients, but each Client has a different number of Authorization Tokens it can construct.

The following paragraphs specify how the token secrets St, Ct and At are constructed, how the tokens can be revoked and how S and C can use their common knowledge to verify the authenticity of the ATs and how to obtain the shared keys VerifK, PSK, IntK, and ConfK.

Construction of Tokens. The main data structure used in this document can be represented as a tree of values. Each value is a Bit string of a fixed size, which is denoted with m. As an example, m can be chosen as 265 Bits. This data structure may be implemented in several different ways, for instance as a set of tables representing the currently relevant parts of the tree.

The tree construction is based on a "root secret", which is denoted by "x", and a Pseudo-Random Generator (PRG), commonly used to generate stream ciphers. In particular, ChaCha20 [Ber08] could be used as a lightweight and efficient algorithm for constrained devices, see [HCFF08]. The ChaCha20 block function can be used as a key-derivation function, by generating an arbitrarily long keystream. ChaCha20 takes as input a 256-Bit key k, a 64-Bit nonce v (could be a unique message number) and a 64-Bit block number. The ChaCha20 output stream can therefore be accessed randomly where any number of blocks can be computed in parallel.

Instead of ChaCha20, other PRG can be used as well, including cryptographic hash functions. With any of those building blocks, it is easy to construct the functions G, g1, g2, g3, g4. Algorithm 1 details the steps.

Algorithm 1 Construction of functions G, g1, g2, g3, g4.

1: **procedure** G(K, I) ▷ The function G maps values from the sets K and I.
2: $G : K \leftarrow KxI$ ▷ K is the key space $\{0,1\}^{265}$ and $I = \{0, 1, 2, ..N\}$.
3: **end procedure**
1: **procedure** G1, G2, G3, G4(K, I)
2: $g1, g2, g3, g4 : K \leftarrow KxI$ ▷ Analogous to the function G.
3: **end procedure**

K is the key space $\{0,1\}^{265}$ (the keys are 256 Bits long) and $I = \{0, 1, 2, ..N\}$ where N is an appropriate integer (a parameter of the construction). In other words, any function G,$g1, g2, g3, g4$ takes an element from the respective sets K (a key) and I (an integer) and maps the output to the set of K again (the output is a key again).

Starting from a secret x, a tree of derived secrets[20] is constructed. The main property of the secrets in the tree is that an attacker can't use the information of a secret to obtain information about other secrets in the tree, except for descendants.

[20]For simplicity, the words "keys" and "secrets" are used indiscriminately.

The knowledge of secrets on the tree reveals nothing about any secret that is not a descendent of any of them.

The children of any node are constructed using a function G ("generator") that takes a key k (of size $m = 256$ Bits) and an index i (the "block number") and creates a new key of size $m = 256$ Bits.

The Token secrets St, Ct and At are all values in the tree and thus can be constructed from x using G. Other functions $g1$, $g2$, $g3$ and $g4$ will be used to generate the derived keys $VerifK$, PSK, $IntK$, and $ConfK$ accordingly.

It is assumed that G, $g1$, $g2$, $g3$ and $g4$ are all publicly known functions.

Main Data Structure. The main data structure used for the PAT protocol may be viewed abstractly as a tree of values, as described above. Each value is a Bit string of a fixed size m (e.g., 256 Bits). But this data structure may also be implemented in several different ways, for instance as a set of tables representing the currently relevant parts of the tree.

For the tree structure, a sequence of integer numbers is used as indexes for the nodes. To avoid parentheses, commas, and semicolons, sequences are written as a string concatenation: "123" is the sequence of three numbers "1", "2", and "3". In order to avoid confusions in all examples, integer sequences do not employ numbers that require 2 or more digits, that is, numbers greater than 9.

The sequences of integers are used to index values in a tree: x_a is the value at the node with position (address) a. In other words, the nodes and their values are denoted as x_a, where a is a sequence of integer numbers. The tree has a root x (also can be written as x_a with $a = \epsilon$, where *epsilon* is the empty sequence). The children of x are x_1, x_2, x_3, ..., x_N, where $k = 1..N$ is a singleton list[21]. If x_a is a node in the tree, then the children of x_a all have the form $x_{a'}$, with $a' = a;i$ where $a;i$ is the concatenation of a and an integer i. The value $x_{a'} = x_{a;i} = x_{ai}$ is calculated as $G(x_a, i)$. Assuming that the output of G is constructed with a cryptographic hash function $h()$, G performs as shown in Algorithm 2.

[21] A singleton list is a list with only one number.

Algorithm 2 Construction of main data structure used by the function G

1: **procedure** G(x_a, i) ▷ Analogous to Algorithm 1.
2: $x_{a;i} \leftarrow f(x_a, i)$ ▷ $f()$ composes the new subindex for a new variable and
3: creates a new value by concatenating the value of x_a with the value of i.
4: $x_{a'} \leftarrow h(x_{a;i})$ ▷ $h()$ is a cryptographic one-way function.
5: ▷ Ideally, $h()$ is an HMAC function.
6: **end procedure**

$f()$ is a fixed (publicly known) function such that for any fixed i the function $f(a, i)$ is one to one. The choice of G should not be regarded a secret: it is a publicly known parameter of the installation for S. It follows that if x_a is known, all descendants of it can be calculated, that is, all nodes in the subtree with root x_a. But not vice-versa: since $h()$ is a cryptographic one-way function[22], the knowledge of $x_{a;i}$ is not enough to calculate x_a. An example is given in Algorithm 3. The example is constructed with the first six characters from the output of the SHA256 function for an arbitrary input that resembles the root x.

Algorithm 3 Example generation of a value for G

1: **procedure** G$(33a209, 1)$ ▷ x_a is a $33a209$, i is 1, a is ϵ.
2: $x_{a;1} \leftarrow f(33a209, 1)$ ▷ $f()$ creates a new variable and assigns a value by concatenating 1 to the value of x.
3: $x_{a'} \leftarrow hash(33a2091)$ ▷ $h()$ is a cryptographic hash function.
4: Return $x_{a'} \leftarrow b21293$.
5: **end procedure**

x_a is read as x sub a or x subindex a. a is called the index or address of the node. Noteworthy hereby is: since also concatenation of Bit strings take place, parenthesis have to be used in that case: $x_{a;i}$ means $x_{(a;i)}$, while $(x_a);bs$ means the concatenation of the Bit strings (x_a) and bs.

A procedure for traversing parts of the tree is needed. For simplicity, a tree of a fixed degree is assumed. That is, each node of the tree has either no children or has exactly a certain amount of children. The notation of i is used as follows: assuming that a "current parent node" x_a and a "current node" (i.e. a child of x_a) exist, the child is written as $x_{a;i}$. Thus, i is an index (an integer) that reveals the child relationship of nodes. For example: the fifth child of x_a is $x_{a;i}$, with $i = 5$.

[22]The properties of a cryptographic one way function are discussed in detail in Section 4.3.2.2.

Traversing the tree with respect to the current parent node x_a, starting at $x_{a;i}$ gives the sequence of nodes (loop) as in Algorithm 4.

Algorithm 4 Traversing the PAT data structure

```
1: Start on node: Cursor ← x_{a;i} and parent node ← x_a
2: while most right sibling of x_{a;i} is not reached do
3:     if x_{a;i} is the right-most child of x_a then
4:         Return value of cursor and stop.        ▷ This will be the output of G.
5:     end if
6:     for each increment of i do
7:         if x_a;i is the right-most child of x_a then
8:             Cursor ← x_a;i
9:             Stop For-loop.
10:        else
11:            Increment i.
12:        end if
13:    end for
14:    if x_a;i has a child node then
15:        Cursor ← x_a;i;1                ▷ Either only child or most left child.
16:    else
17:        Cursor ← x_a;i
18:        Return value of cursor and stop.        ▷ This will be the output of G.
19:    end if
20:    if x_a;i;1 has a right sibling node then
21:        Cursor ← x_a;i;1 and parent node ← x_a;i
22:        Repeat While-loop.
23:    else
24:        Return value of cursor and stop.        ▷ This will be the output of G.
25:    end if
26: end while
```

Construction of St, Ct and At. The secret x is the main secret. It is generated by the SAM as a random or pseudo-random number of m Bits (m is taken to be 256 Bits). The method used to construct x is out of scope, but it should be practically impossible[23] to guess by an attacker, even if he knows a sequence of previous or future choices of x.

[23]The notion *practically impossible* is used here to denote that an attacker would need a considerable amount of time (e.g. years), computing power and/or economical resources to feasibly attack the used mechanism.

Algorithm 5 Initial creation of St

```
1: procedure St(x)
2:     St ← x                                   ▷ Initially, St = x.
3:     Return St
4: end procedure
```

St is generated initially as shown in Algorithm 5 and is sent by the SAM to the Server S in the message 3 of Figure 4.15. The value of St at the Server may change if the current value of St is revoked by the SAM.

The root has N children, one for each foreseen Client. The value x_i, for $i = 1..N$ is a secret associated to Client number i, but it is not known by the Client. The values of the nodes x, $x_1, x_2, .., x_N$ are secrets that never leave the SAM or the Server S and should not be leaked. The first children of $x_1, x_2, .., x_N$ are the initial values of Ct. In other words, for Client number i:

Algorithm 6 Creation of Ct

```
1: procedure Ct(x, i)
2:     x_i ← h(f(x; i))
3:     Ct ← x_{i;1} ← h(f(x_i; 1))              ▷ Initially, Ct = x_{i;1}.
4:     Return Ct
5: end procedure
```

Ct is generated according to Algorithm 6 and sent by the SAM to C(i), where i is the Client as in message 7. Also in this message, the SAM sends the "current node" (used by C to start a traversal), the depth and the degree of the sub-tree at the node Ct.

The value of Ct at the Server may change if the current value of Ct is revoked[24] by the SAM. If this happens, it is necessary for the Client to obtain a new Ticket Grant (message 7). To create the sequence of Authentication Token secrets, $At1$, $At2$, ..., the Client traverses the tree according to Algorithm 4 starting at a "current node" (determined by the SAM in the parameters of the Ticket Grant at message 7) with the current parent node being the current value of Ct.

Formal verification and evaluation of efficiency. Chen models in [Che15] the PAT protocol (draft version 01) using the Scyther language[25], see [Cre14a] for

[24] Details on revocation are planned for a future version of the protocol.
[25] A deeper insight into Scyther is given in the Section B.2.

the language's formalization details. The Scyther model does not fully represent the protocol due to the short comings of the Scyther language (e.g., state conditions cannot be properly modelled such as changing the cursor in the data structure as shown in Algorithm 4.) Therefore, Scyther identifies two critical vulnerabilities that exploit replay messages for the Client and the Server. Chen has formally analysed the shortcomings of the Scyther model and shows that the protocol is indeed not vulnerable to the identified attacks.

Also, an evaluation of the efficiency and power consumption of several Authorization and Authentication (AA) schemes for constrained[26] devices are found in [Che15]. Chen measures the energy consumption of different cryptographic algorithms (SHA256, Advance Encryption Standard (AES), Triple-Data Encryption Standard (3DES) and Elliptic Curve Digital Signature Algroithm (ECDSA) Sign and ECDSA Verify) that are needed by the respective authorization protocols (Kerberos → AES or 3DES and SHA265, Elliptic Curve Qu-Vanstone Implicit Certificates (ECQV) → ECDSA Sign and ECDSA Verify, and the PAT protocol → SHA265 or CHACHA20 on a MSP430 LauchPad with a MSP430F5529 micro-controller [Ins14], which has 8KB RAM and 128KB Flash. The micro-controller has no hardware accelerator for any cryptographic algorithms. This is a key aspect as discussed in the evaluation of economic constraints[27]. The cryptographic functions are provided by the open source cryptographic library of the operating system RIOT [HBG+13]. In the evaluation, the consumption profile of the 802.15.4 protocol (send and receive) is contained as well. Figure 4.16 shows the comparison.

[26] The constraints of Sections 2.6.2 and 2.6.1 apply to Chen's evaluation indiscriminately.
[27] Detailed in Section 2.6.1.

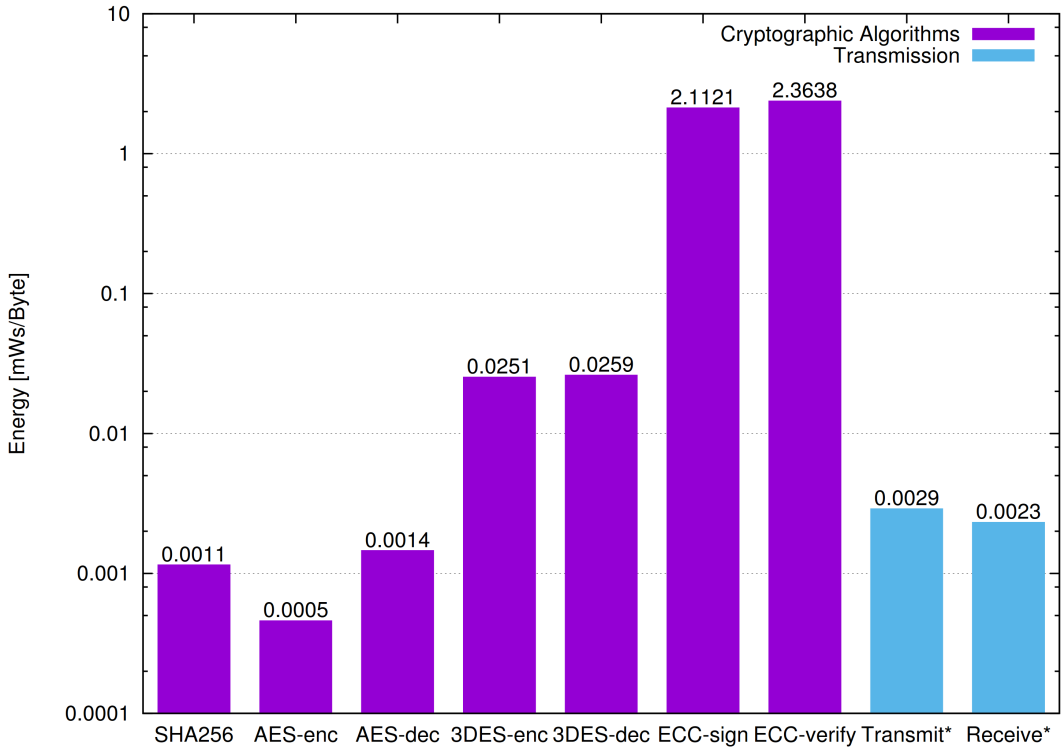

Figure 4.16: Energy profile comparison of different cryptographic functions [Che15]

The values shown are relative values in mWs/Byte (1 mWs = 1 mJ for comparability with the battery capacity listed in Section 2.6.2). Because the processed block sizes of the cryptographic algorithms differ, the results are divided by the number of the respective operated Bytes: SHA256 operates with 32 Bytes, AES with 16 Bytes and 3DES with 8 Bytes block of data. ECDSA with the specific curve $secp160r1$ (a 160-Bits curve) signs and verifies 20 Bytes of data.

The evaluation contains three types of algorithms: hashes, symmetric encryption and asymmetric encryption. Notably, symmetric (AES, 3DES) and hash algorithms (SHA265) are much less energy consuming than the most efficient asymmetric counterpart (ECC), namely up to a factor of four thousand (4727,6). The factor may considerably affect the battery lifetime of constrained devices and enable security and privacy in use cases that could not have been possible before. An additional analysis of the computational consumption can as well be found in Chen's work [Che15].

As mentioned, the energy profile of the data transmission is also plotted (light

blue bars in the chart). The corresponding measurements used a CC2530 network processor (2.4 GHz IEEE 802.15.4) [cc2].

Chen has further broken down the protocols by their cryptographic operations and message sizes to generate an overall computational and power consumption profile:

Algorithm	Average Message Size in Bytes	Energy Consumption per Message (mJ)	Memory Consumption in Bytes	States needed (for n active Clients) in Bytes
ECQV (ECDSA)	40 + 114	47.92	> 6858	56 * n
Kerberos (AES)	40 + 48	0.56	> 43262	32 * n
PAT (SHA256)	40 + 32	0.53	> 1450	32 + n

Table 4.1: Comparison of energy and computational profiles of AA schemes

Table 4.1 compares the profiles of all three AA schemes. Overall, the PAT protocol has the lowest energy and computational consumption profile. The symmetric schemes are more efficient by a factor of approx. 2 for message size and a factor of approx. 88 in energy consumption per authorization request message. Note that the authorized request message includes generation (usually by the Client) and verification (usually by the Server) efforts. If both symmetric schemes are compared, the PAT protocol has a much better computational consumption than Kerberos due to the use of SHA256. Also, the burden of the Server is much lower in the PAT protocol, as the Server has to remember only one 32 Byte hash value and concatenate the n integers (the subindexes) to construct the keys for various Clients. This is not the case in Kerberos, as the Server has to remember a respective key per Client (thus 32 Byte * n).

It should be noted that with the ChaCha algorithm family, the computational and power consumption profile can be improved by a theoretical factor of five to twelve. This alternative is proposed in the draft of the PAT protocol itself and in [Che15, pages 49, 115]. Table 4.2 summarizes Chen's evaluation of AA schemes for constrained devices in lossy networks.

Algorithm	ECQV	Kerberos	PAT
Performance	- ECDSA consumes considerably more energy.	+ 3DES neither energy efficient nor considered secure. The AES implementation consumes a non-negligible amount of flash memory.	++ SHA265 and AES are applicable. SHA256 has a low energy and computational consumption profile.
Scalability	++ Based on the asymmetric cryptographic algorithm ECDSA.	+ Session keys are managed by a non-constrained Kerberos Key Distribution Center. Server has to memorize one key per client (32 * n Bytes).	++ The data structure allows the Server to memorize only 32 + n Bytes. Additionally, the delegation of authorization and authentication management from the Server to the SAM supports scalability.
Key Generation and Agreement Complexity	- The public key of a Client must be calculated using the public key of the CAM. The CAM must be able to mime a Certificate Authority in a PKI environment, which is arguably difficult. Agreement between parties is efficient.	+ A RS is assigned a Kerberos ticket, which is an encrypted set of keys. The RS verifies the acquired ticket by means of decrypting the ticket and optionally testing a keyed message authentication code.	++ Used secrets are agreed on by only exchanging few integers (sub-indexes).
Revocability	- Certificate Revocation List grows with time considerably.	- Revocation has considerable management effort and influences overall possible ticket lifetime.	- Method detailed in a further version of the draft.
Maturity	+ Used in ZigBee networks, but no open source implementation can be found.	++ Open source and reference implementations exits.	- Formal verification exists but no reference implementation.

Table 4.2: Evaluation of AA schemes for constrained devices in lossy networks

Where "++" means the evaluated property is fulfilled, "+" means partially fulfilled and "-" means not fulfilled.

4.3 Privacy Enhancing Technologies supporting Confidentiality

In this Section, privacy enhancing technologies supporting confidentiality are introduced. This technologies have data minimization by design and try to circumvent intentional or unintentional hoarding of personal data and the aggregation and linkability of user information through several systems.

The technologies presented in this Section support all Rerum use cases: UC-O1, UC-O2, UC-I1 and UC-I2 [SWC+15]. They are all part of the Rerum security, privacy and trust technologies, implemented solely on the Rerum devices.

In this Section, the technologies will be presented individually and sequentially integrated into a final IoT domain model in Section 5.

4.3.1 Position Hiding in Floating Traffic Observation

Geo-Location PETs support data minimization in traffic applications. They enable the system to send the minimal amount of information to location-based service providers. This is of significant importance, as the tracking of location information discloses a large amount of information about the habits, activities and preferences of users.

In this Section[28], a location privacy PET for floating car observations will be presented. As this PET is tailored to Rerum's mobility use case, a short description of the use case is given in Section 4.3.1.1.

4.3.1.1 Floating Car Observation

Smart transportation is one of the main scenarios of Rerum. The geo-location PET was tailored for Rerum's use case, although it can be applied in any floating car observation scenario. It should be noted that floating car observation differs

[28]The contents of this Section have been previously published for Siemens AG as an intellectual property item (number 2014E07575DE) and for Rerum in [RER14b, SWC+15].

from many other smart mobility use cases, such as Vehicular Ad-Hoc Networks (VANETs).

Floating car observation is a variation of traditional traffic flow monitoring. Periodic monitoring serves as the foundation for road construction and traffic planning. Planning includes e.g. variation studies, traffic forecasts, traffic and road engineering and accident statistics. The German Federal Highway Research Institute (BASt) states that the most important data for this purposes is (i) the type and amount of vehicles in traffic, (ii) weight and speed, and (iii) the route taken [Ins15].

Traditionally, stationary devices record the needed data. Selected parts of high ways and federal roads are equipped with data collection systems. Additionally, manual traffic monitoring is carried out on regular intervals.

In order to increase the granularity of that data (speed, amount and current route that is being taken), particularly routes that are less frequented, digital floating car observation can be implemented. The digital acquisition envisions the use of an ubiquitous GPS module on every traffic participant [STBW02]. Figure 4.17 visualizes the data that is gathered in a GPS-based traffic measurement setup.

In the simplest scenario, apart from a user's IP-address, information about a user's exact position in certain points of time is transmitted. Figure 4.17 shows measurements M1 and M2 as an example. In every measurement, the user's position is given as a latitude/longitude set, including the time of the measurement. This is enough to calculate a vehicle's speed (difference of distance and points of time between measurements) and direction (difference between locations), as well as to disclose information about the user's current route and his daily points of interest (by mapping of measurements to geo-databases).

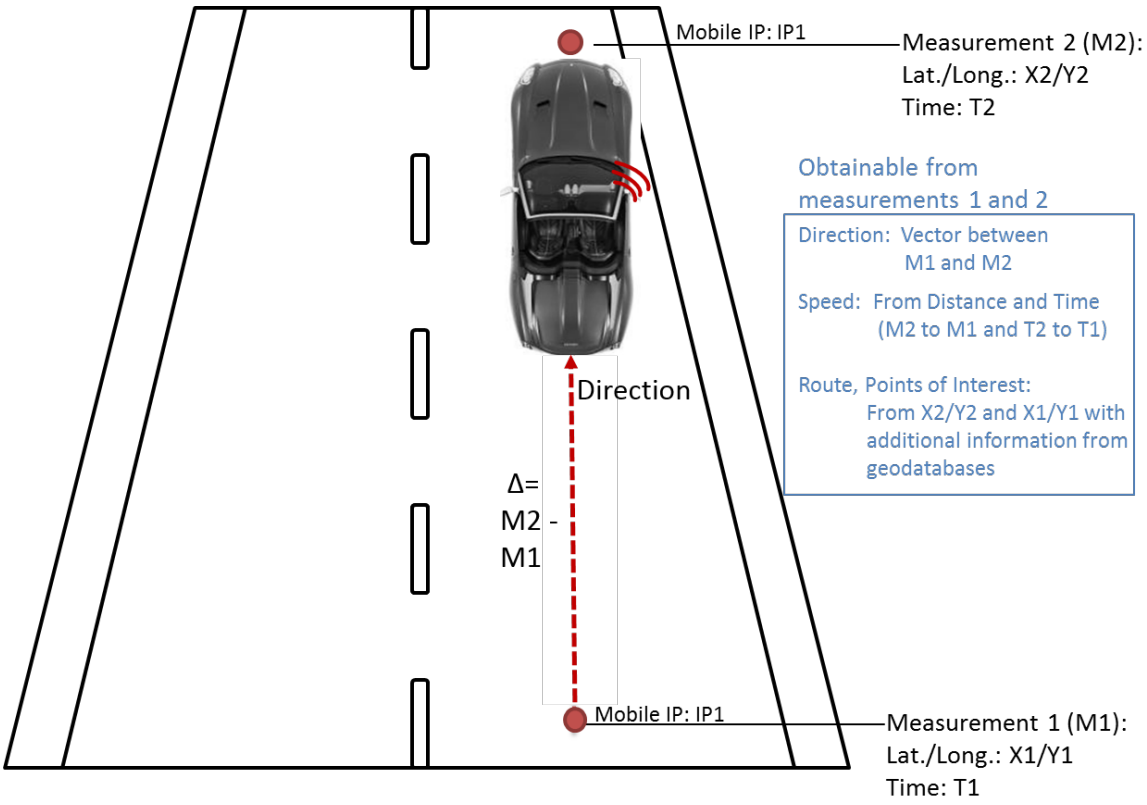

Figure 4.17: Geo-location data in traffic measurement

The measurements have to be sent in such a way, that the sending participant cannot be linked to the data sent to protect him of intrusive conclusions about his daily habits. Depending on the scenario, measurements will contain information about a certain street/route/location and the average speed, the measurements are transmitted close to real-time.

The main idea behind the PET is to pre-processes and aggregate measurements, omit detailed coordinates and hide the user's identity. The resulting scheme should still allow the service provider to obtain the needed information for traffic analysis and forecast.

4.3.1.2 Related Work

Many proposals to avoid tracking in vehicular environments exist, such as [BS04, HMYS05, BHWW09]. To understand why floating car observation scenarios are not covered by currently proposed geo-location PETs, the existing approaches have to be categorised. Most geo-location privacy approaches hide traffic participants

in vehicular networks, where messages from vehicles are routed through traffic participants, using other vehicles as routing nodes. In a such a simplified vehicular network, a traffic participant requests nearby vehicles to route its message over the network to hide itself as the source. In more advanced scenarios, for example noted in [BS04], the vehicles broadcast sets of their positions, speeds, motion vectors and acceleration as so called Beacons every 100 to 300 milliseconds. Mechanisms in VANETs protect these Beacons and other VANET messages by hiding the vehicle's identity with pseudonyms and obfuscating the sending routes. Similar to mix-cascades and onion routing for network traffic, VANET privacy mechanisms use mixing of message routes and identities, creating so called mix-zones, as described by Beresford [BS04]. There are more variations of the mix-zone concept including trusted third parties and pseudonyms. An extensive overview is given by Scheuer in [SPF08].

In floating car observation, the situation is different. A traffic participant does not need other participants to broadcast his message. The traffic data measurement is transmitted directly to a service provider, possibly using a cellular mobile network (e.g. 3G or 4G). As the network transmission can continuously identify the participant, anonymous routing techniques have to be applied. This will not be a research focus, as many applicable anonymous network solutions exist such as the TOR [MBG+08] and the AN.ON [Fed07] networks.

In addition, most VANET privacy mechanisms protect message routing, but not the message content itself. The message content, i.e., the measured GPS positions and driving speeds, is the privacy sensitive data in a floating car observation use case. Therefore the need to identify suitable techniques for transmitting detailed traffic information is evident, but at the same time the traffic participant as a data source has to be protected. This is done by enlarging the set of indistinguishable measurements.

Every traffic participant simulates not one, but several participants sending measurements. As the number of simulated participants is generated randomly, the anonymity set varies in such a manner that the change of distinguishing single participants from simulated or other real participants becomes insignificant. At the same time, the measurement data is left unaltered. There is no aggregation

or perturbation of measurements for the service provider. A detailed description of the mechanism is provided in the next subsection. Also, privacy considerations are given in Section 4.3.1.9.

It should be noted that efforts exist to formally verify VANET schemes. The verification is based on tool support, for example with the VANET simulator [TSF12]. The VANET simulator simulates traffic flow of several participants and analyses different tracking possibilities with and without PETs. The scope is different for floating car observation, thus making it necessary to modify and occasionally maintain an own branch of tools.

4.3.1.3 Description of the Scheme

This Section formulates the technical details of this privacy enhancing technology. The scheme allows traffic analysis by floating car observation and the adoption of user preferences and temporary opt-out of the data collection. For example, a user might not want to send traffic data for a specific area. But as soon as he passes it, it's fine for him to participate in the data collection again. The presented geo-location PET allows this kind of situations.

Additionally, privacy-by-default approach is adopted and the data collection is stopped at side-roads, which are less affected by heavy traffic.

4.3.1.4 Generation of Vectors

The main data structure of the geo-location PET are vectors. The vector creation is illustrated in Figure 4.18.

A vector is created the following way: when a user is moving, a timer decides where the starting point of a vector will be, and how long it will take to choose the ending point of the vector. As several vectors may be created at the same time, the traffic participant will have a list of current vectors.

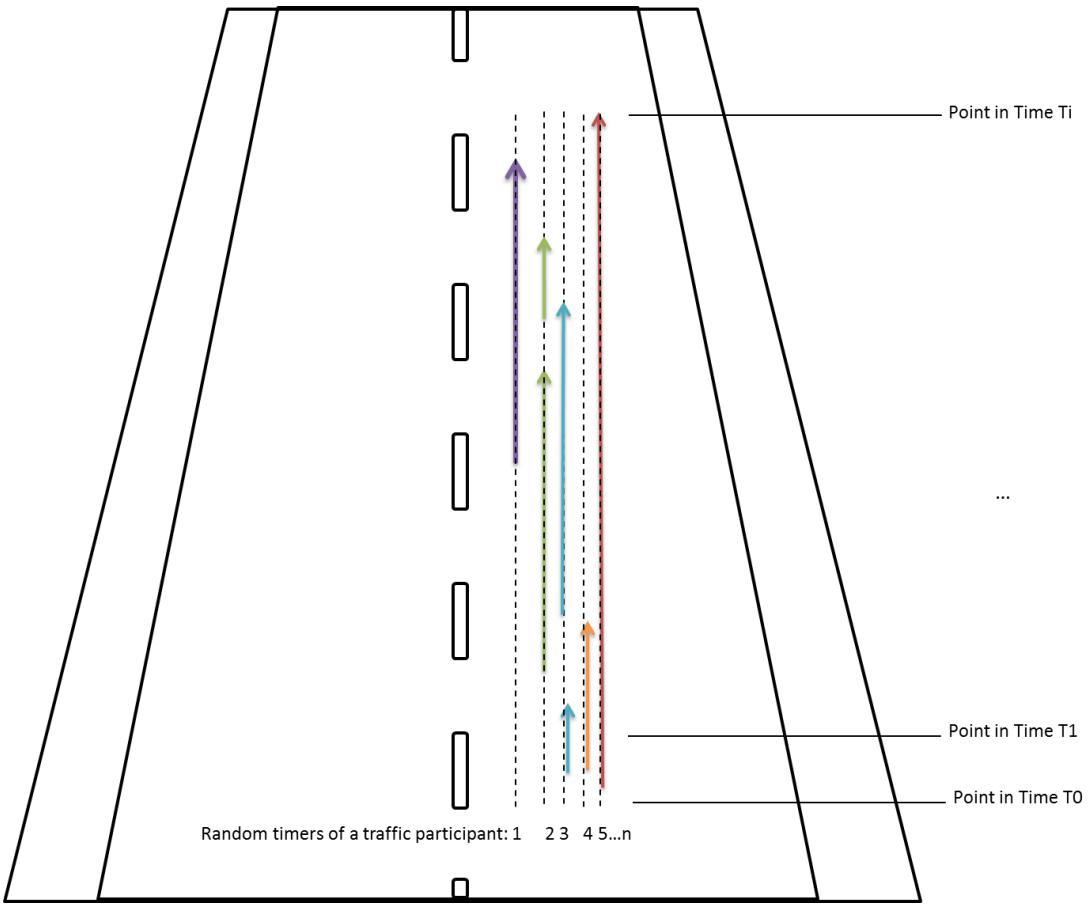

Figure 4.18: Time controlled vectors

Figure 4.18 illustrates the vector creation. An example list of current vectors is given in Table 4.3.

Vector	Starting Point	Time Until Stop	Average Speed	Average Driving Time	Endpoint (Elicited at Stop)
A	(X11, Y11)	5 Minutes	50 KM/H	21 Minutes	(X12, Y12)
B	(X21, Y21)	8 Minutes	...	...	(X22, Y22)
C	(X31, Y31)	22 Minutes	..	...	(X33, Y33)

Table 4.3: Generation of multiple vectors

The "starting point" and the "time until stop" are chosen at random. The endpoint is measured at the moment when an assigned timer runs out. Afterwards the vector information is sent to a service provider, e.g., the traffic department. While the amount of vectors prevents the knowledge of how many participants are really passing the same route (each vector is transmitted as a unique traffic

participant), the attached speed and driving time averages provide information about the overall traffic of each route. The vector creation and transmission aim at an artificial set of indistinguishable participants and data sources and thus create a k-anonymity set. Further privacy considerations are formulated in Sections 4.3.1.8 and 4.3.1.9.

The scheme is designed to stop the data collection as soon as either a policy specified location or a side-road is reached, and can be used to support privacy location policies as well, such as [STC+13]. Side-roads and areas defined in privacy location policies are opt-out areas, as the user (automatically) opts-out of the generation of vectors and the transmission of traffic data. Two actions to support a participation opt-out are defined:

The first action is *stop at geodic/civic location condition*, which stops the data collection and transmission while the participant is in a defined area, and the second action, *stop at side-road*, which stops the data collection whenever a traffic participant exits a main road and enters smaller side roads.

Smaller roads lead to a participant's home, working place, etc., and thus are, in the presented opt-in approach, excluded by default from the analysis. The stop behaviour is as follows: several independent vectors are generated and sent to the a service provider at random as usual, with the addition that the generation of vectors will stop when the participant's policies apply or when he enters a side-road.

To exemplify the different actions, a traffic participant driving in Regensburg, Germany is assumed. The participant has defined policies of a location (green circle in Figure 4.19), where the data collection system should stop.

4.3.1.5 Stop at Geodic/Civic Area defined by Policy

It is assumed that a traffic participant has defined some areas where he does not want to send traffic information. One way of defining such policies is by using the geodic and civic location profiles described in RFC6772 [STC+13]. The interpretation of such a policy has been done before, see [DW10], and is thus

not a part of this proposal. The traffic participant generates random vectors as described in the Section above, see 4.18; as soon as he reaches the defined area the generation vectors will stop. Active vectors will be sent to the traffic department.

Figure 4.19 exemplifies the reaction of the system when the "stop at geodic/civic area" action is defined. The traffic participant has defined a policy to opt-out when he reaches his residential area around *Friedrich-Ebert-Strasse* (green circle), which is assumed to be an area of social flashpoint. The route he takes is depicted by the black dotted line; the vectors generated throughout the route are of several colours.

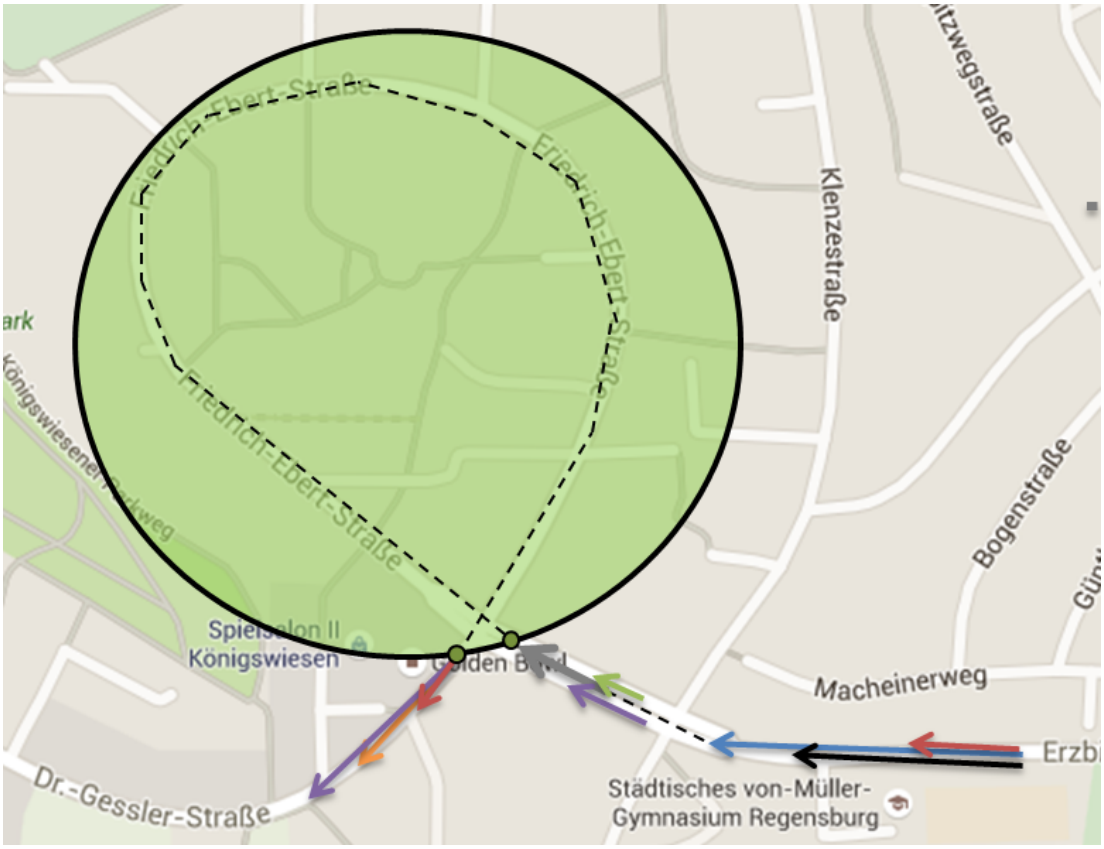

Figure 4.19: Stop at geodic/civic area

At reaching the circle, the data collection will behave as follows: no vectors will be generated starting from this point, and, if any active vectors exist, a common average will be generated and sent as a position somewhere before the entry point to the protected area. A detailed example of how averages can be generated is given in Tables 4.4 and 4.5.

4.3.1.6 Automatic Stop at Side-Roads by Default

The automatic stop at side-roads by default is exemplified in Figure 4.20.

Figure 4.20: Stop at side roads

The user's route is depicted as a black dotted line. The user drives along *Erzbischoff-Buchberger-Allee* and enters *Friedrich-Ebert-Strasse*. He then decides to take detour at a small side-road along (depicted as a a black dotted line traversing Friedrich-Ebert-Strasse). At the point of entrance (small black-framed green circle), the data collection will stop. This means, that no vectors will be generated starting from this point. If any active vectors exist, a common average will be generated and sent as a position somewhere before the entry point of the side-road.

4.3.1.7 Example of User Opt-Out with two Active Vectors

Let's assume two vectors are still active while the user enters the side road (e.g., the green vector in Figure 4.20).

Vector	Starting Point	Time Until Stop	Average Speed	Average Driving Time	Endpoint (Elicited at Stop)
A	(X11, Y11)	5 Minutes	49 KM/h	1 Minute	(X12, Y12)
B	(X21, Y21)	8 Minutes	30 KM/h	7,5 Minutes	(X22, Y22)

Table 4.4: Multiple active vectors before entering an opt-out area

The two vectors will be averaged, converted to one vector, an endpoint is assigned that differs from the entry point of the side-road and sent to the service provider.

Vector	Starting Point	Time Until Stop	Average Speed	Average Driving Time	Endpoint (Elicited at Stop)
A	($\widetilde{X1}$,$\widetilde{Y1}$)	-	39,5 Km/h	4:38 Minutes	($\widetilde{X2}$, $\widetilde{Y2}$)

Table 4.5: Averaged vector sent at entrance of an opt-out area

The new vector (shown as a two-lined gray vector in Figure 4.20) has an endpoint with a GPS-position somewhere on the dotted black line, before the entry point to the side road. This is the last vector sent before the participant enters the side road. After leaving the side-road, the participant starts sending position data again; this is denoted by the red, orange and purple vectors in Figures 4.19 and 4.20.

4.3.1.8 Avoiding Correlation between Vectors

To avoid a possible time correlation between the last averaged vector, the driving speed and a new vector, a random threshold time until opt-in is suggested. Thus the correlation between the new vectors and the previous vectors through the driving speed is blurred.

4.3.1.9 Privacy Considerations

As described previously, every vector has to be sent as a unique traffic participant's measurement. To achieve this the IP-address from the sender has to be hidden with anonymization techniques such as TOR, and if needed, adding integrity

protection in form of unlinkable group signatures [CVH91]. As seen in [CML06], merely protecting the sender of GPS-location data is not enough.

Additional information, for example by a geo-location system and online social networks, reveal where the traffic participant is heading to and which users are the ones that could have possibly visited those locations. The set of these subjects, or the k-anonymity set of data subjects where each participant is indistinguishable from at least k-1 other participants with respect to a certain GPS-position, is often very small. The reason for this is, that every GPS-location can be linked to driving speeds, time correlations and locations, thus the resulting sets become very unique.

The generation of artificial vectors enlarges that anonymity set, but without blurring or adding noise to measurements. The artificial vectors provide even more information to the measurements, as every vector has its unique starting and ending point.

It should also be noted that vectors are safe of correlation by time and driving speed, if the driving speed of participants is assumed similar. In case of pedestrians or cyclists, the vector speeds could be significantly different and could reveal which vectors where generated by which users. Thus the vector based GPS-location privacy technology is only suitable for participants with similar motion speed.

4.3.1.10 Summary and Future Work

The geo-location privacy component is a novel approach to privacy friendly floating car observation. Related work on vehicular area network has focussed on hiding message routes, while it does not analyse GPS positioning data. The proposal on random vector generation intends to fill this gap and allow for accurate measurements for service providers as well as location and policy based privacy for users. Additionally to the trial use case of Rerum, the formal verification of the random vector scheme with tools like the VANET simulator [TSF12] or other tools is considered as future work[29].

[29] As mentioned before, most tools centre on VANETs and do not cover floating car observation. Therefore, verification requires therefore a considerable effort on tool modification.

4.3.2 Pseudonyms

A pseudonym system supports data minimization by hiding the identities of devices and users from the services and other system participants, if they are not necessarily needed. In cases where attackers or intruders are able to steal records from databases, pseudonyms will prevent that individuals are tracked down through their identities.

This Sections presents a PET for pseudonym generation, agreement and management[30]. The scheme is designed to work under the limitations scoped in Section 2.6.2. The scheme uses only symmetric cryptographic mechanisms, particularly one-way functions, as they have proven to be efficient in computation, energy consumption and code size, see Section 4.2.5.1.

4.3.2.1 Related work

Pseudonym systems can be categorized in three concepts, see [TPT06]. Spatial concepts are based on mix-zones, where pseudonyms are exchanged when the pseudonymized participants meet. Time-related concepts change pseudonyms after a certain time. User-oriented concepts allow the users to decide when their identity should be changed. The decision can be based on the user's own policies and thresholds for the automatic pseudonym change.

All this concepts have in common that they concentrate on the management of when and why pseudonyms should be changed but not directly on how the pseudonyms are created. This concepts are complementary to the proposed pseudonym PET.

Other proposals rely on asymmetric schemes which do not allow obvious re-linking of signatures to their cryptographic source material [LRSW99].

The presented pseudonym generation and management mechanism is based on Hash-Trees, similar to those found in the Merkle-Signature-Scheme (see [Mer79]). Merkle trees are generated in a bottom-up approach: every element of a set with $nmod2 = 0$ values is hashed to generate n different hash values. The elements are paired, their hash representations are concatenated and hashed to get a new

[30]The contents of this Section have been previously published as a Siemens AG intellectual property item (patent numbers 2015P01590 DE, 102015203543 DE, and 102015203543 A1) and for Rerum in [RER14b, SWC+15].

set with $n2$ hash values. This operation is repeated recursively with every new set of hash values until $n = 2$, thus the final set resulting in one single hash value, called the top hash or, figuratively, the root of the tree. Merkle trees are static, thus used e.g. to validate file integrity on IT systems. Also, a public key infrastructure has been proposed with Merkle trees [Mer90], although the proposal is practically inefficient due to the dependency of the public key to fixed and pre-signed messages.

In contrast to Merkle trees, a top-down approach is proposed, which allows generation of practically infinite hash values which are used as pseudonyms.

The generation of tree structures containing values and their generation with one-way functions have been also proposed by Goldreich, Goldwasser and Micali [GGM86]. The formal tree structure resembles that of the top-down hash-trees proposed in this thesis, but they differ in form of generation of values: the Goldreich, Goldwasser and Micali (GGM) tree uses a single root value x that is passed to several different functions subindexed $G_0x...G_nx$. The generated output is a practically infinite random string of values, depending on G_ix. The top-down hash-tree uses sub-indexes and particularly a binary set of 0 and 1 mainly as input and traversal information. The root value x is regarded as the key anchor, is only used once, and is kept as little as possible flowing in the system.

Weis et al. [WSRE04] have proposed a pseudonym protocol for RFIDs where the IDs of an RFID-Tag are hashed. The reader brute-forces the received pseudonyms by hashing all its known IDs (hash-functions of tag and reader are previously agreed on). This protocol is computationally heavy on the Verifier (which could be a constrained resource server in IoT) and allows a re-linking mechanism that is not applicable in Rerum's use cases.

Ohkubo et al. [OSK+03] introduce a scheme for RFID pseudonyms based on one-way functions (hashes) and a counter. The counter increases with every emitted pseudonym, the same counter is used to keep track of valid pseudonyms. Again, this protocol does not protect the claimer (tag) from the Verifier (reader).

The scheme by Juel [Jue04] uses an XOR operation and a secret value table. The problem with this scheme is that only a limited number of pseudonyms can be generated before the table has to be refreshed by a trusted third party. Molnar

et al. [MW04] use a hash-tree structure to achieve mutual authentication between a RFID-tag and a reader. The scheme protects the tag from being tracked by third parties. In an IoT use case, the RFID claimer (tag) must be also protected from the Verifier (reader). This is not scoped by Molnar et al.'s protocol. Another proposal by Molnar et al. [MSW05] uses a similar setup, where again an RFID-tag is protected by a pseudonym. Different than before, the tag is protected against the reader, thus closing the gap between a typical RFID and IoT scenario. Molnar et al. use here a tree of values as well. One single tree is used to generate more RFID tags (a method that is also proposed in the following PET), but the generation differs in the way the values are created. Molnar et al.'s proposal use a pseudo-random generator and a counter to generate the values in the form of a GGM tree, where the input value and a counter are the same input for all the generated pseudonyms, as proposed in the GGM tree structure. That means, that the input value is ubiquitously used in the scheme (starting from the $d1$ level in [MSW05]), and if stolen, can be used to disclose the pseudonym stream from start to end.

The pseudonym PET of this thesis proposes a small but important change in this regard, as the root input value is only used once to support perfect forward secrecy. Another difference lies on the tree traversal that allows conditional pseudonym agreement on different ownership situations. Molnar et al. target a similar mechanism with delegation, but either a root value or a large number of values would have to be published.

The following pseudonym PET aims to provide an efficient way to generate practically unlimited pseudonyms, provide perfect forward secrecy for one or a set of disclosed pseudonyms, protect claimer and Verifier and allow temporary pseudonym agreement without re-linking the real identities of a claimer.

Firstly, the fundamentals of one-way functions and the tree data structure are recapitulated. Secondly, the scheme is built up and mechanics to reach the described goals are sketched. Finally, the scheme is integrated to the Rerum domain model.

4.3.2.2 Existing Fundamentals

In the following Section existing fundamentals for the creation of top-down hash-trees are rounded up.

One-Way Functions. An one-way function is a function *f()*, which takes x as an input and computes y as an output. Computing y as an output is hereby easy, while computing x from y and *f()* is practically impossible.

Hash-functions. A hash-function is a special type of an one-way function *h()*, which takes the input set X containing binary coded elements of any length, and produces an output set Y of binary coded elements with a certain length n, where following properties apply [NY89]:

One-way or non-invertable property: it is virtually impossible to compute $x \in X$ from $y \in Y$ and the hash-function *h()*, where *h(x)=y*.

Collision resistance: it is very unlikely to find two (or more) inputs $x_1, x_2 \in X$, where $h(x_1)=y$ and $h(x_2)=y$.

Chaos: even similar inputs generate significantly different outputs. Changing an input by one Bit should generate and output that is about 50% different than the output of the unchanged input.

Keyed-Hash Message Authentication Code (HMAC). Keyed-Hash Message Authentication Codes are used here, defined in RFC 2104 [KBC97]. HMAC has been designed to generate keyed message authentication codes by applying well understood cryptographic hash functions with a shared secret, see [KBC97]. It should be noted that this is not the only method of how to use hash-functions with shared secrets and that the selected hash function method is irrelevant for the rest of the approach.

A Keyed-Hash Message Authentication Code (HMAC) is a specific construction for calculating a Message Authentication Code (MAC) involving a hash function in combination with a secret key. As with any MAC, it may be used to simultaneously verify both the data integrity and the authentication of a message. Any hash function may be used in the calculation of an HMAC. In RFC2104 [KBC97], an HMAC is calculated the following way:

Algorithm 7 Generation of HMAC values

1: Start with a message m, a hash function $H()$ with the block-size B and a key K.
2: **if** $|K| < |B|$ **then** ▷ $|x|$ means the length of x.
3: create *ipad* such that the $|K + ipad| = |B|$
4: ▷ *ipad* is a one-block hexadecimal constant with the value $0x36...36$.
5: create *opad* such that the $|K + opad| = |B|$
6: ▷ *opad* is a one-block hexadecimal constant with the value $0x5c...5c$.
7: **end if**
8: create $HMAC \leftarrow H(< K \oplus opad, H(< K \oplus ipad, m >) >)$
9: ▷ The operator $< a, b >$ denotes the concatenation of a and b.

$H()$ is a cryptographic hash function with an input block size of B. Cryptographic means here, that the function has the properties described above (one-way, collision resistant, chaos property). K is a secret key padded with extra zeros to match the input block size of the hash function. If K is longer or the same size as B, no padding is needed. m is the message to be integrity protected and authenticated. () denotes a function input, $<,>$ denotes concatenation. $\oplus$ denotes the XOR operation. *opad* is the outer padding (0x5c5c5c...5c5c, one-block-long hexadecimal constant). If K is smaller than the block-size B used by the hash-function, this padding extends the key to that length. *ipad* is the inner padding (0x363636...3636, one-block-long hexadecimal constant). This works the same as the opad, but with a different value.

The HMAC scheme uses two hash operations to generate the message authentication code. The hash-operation count will be important in later stages of the protocol to verify the computational and power efficiency.

4.3.2.3 Virtually Unlimited Generation of Values

The main idea of the PET is to use the output y_i from hash-functions as pseudonyms: due to the one-way property, it is practically impossible to invert x from the publicly known pseudonym y and the used hash-function *h()*. Due to the chaos property, it is possible to compute two pseudonyms from a slightly different value x and use this outputs again for the generation of other pseudonyms, which allows generating virtually unlimited pseudonyms from one initial value. The generation and coordination of values is based on aforementioned top-down hash-trees:

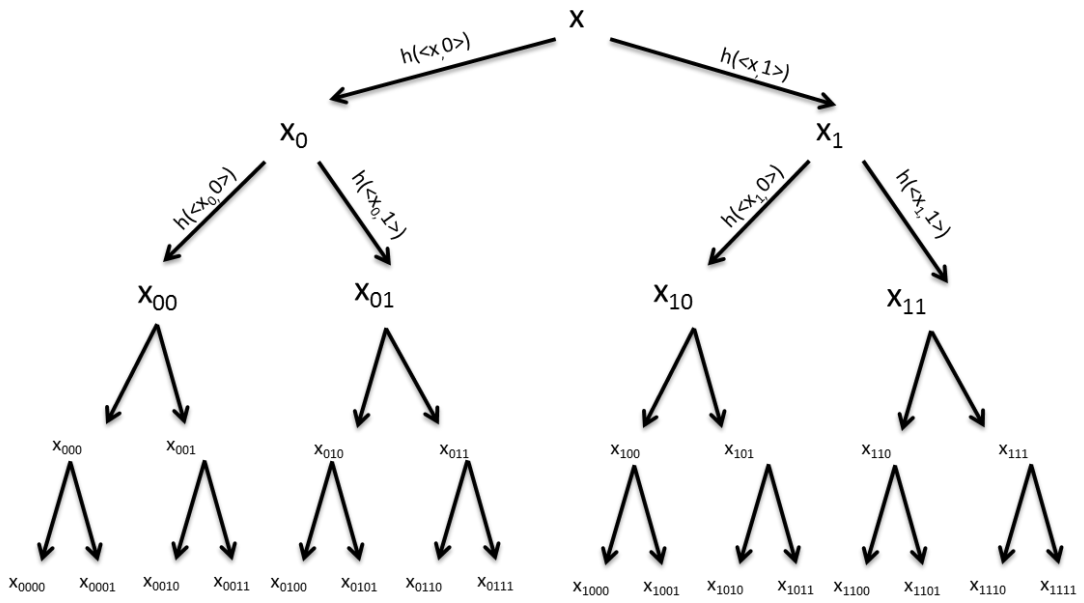

Figure 4.21: Hash-Tree with an initial input x

Figure 4.21 illustrates the steps needed to create a hash-tree. An initial input x is represented as a binary sequence. It is the seed for the generation of all other values. How the initial input x is obtained, can be very different. It might be from an authenticated Diffie-Hellman-exchange, a hashed-password known to two or more parties, etc. This is irrelevant for the rest of the approach. The input x is concatenated with one additional Bit, "0" and "1", respectively, and given to the hash-function $h()$. The outcome is two outputs x_0 and x_1 with length n (depending on the hash-function), which in turn are going to be used as inputs for the next branches. The used hash-function and the generated lengths for the outputs x_i can vary; every hash-function with the properties described above (non-invertible, collision resistant, chaotic) can be used for this approach. In the next step, x_0 and x_1 are again concatenated with one additional Bit, "0" and "1", respectively. They are used as inputs for the hash-function $h()$, which again generate two outputs each, namely (x_{00}, x_{01} and x_{10}, x_{11}). By repeating this step several times, a virtual infinite hash-tree can be build. Note: Figure 4.21 reuses the notion introduced in the explanation of HMACs, where $<,>$ denotes concatenation and $h< x_i, 0 >$ denotes, that the concatenated input of x_i with "0" is given to the function $h()$.

4.3.2.4 Choosing Adequate Pseudonyms

As noted above, due to the one-way property of hash-functions, outputs could be used as publicly known pseudonyms, without revealing the input from which they were generated. Once an output is publicly known, it does not qualify as an input for the generation of other pseudonyms. Thus, a path has to be chosen which allows using outputs as new pseudonyms and at the same time allows generating new branches of pseudonyms nonetheless. Many paths exist, in this thesis the path shown in Figure 4.22 is proposed:

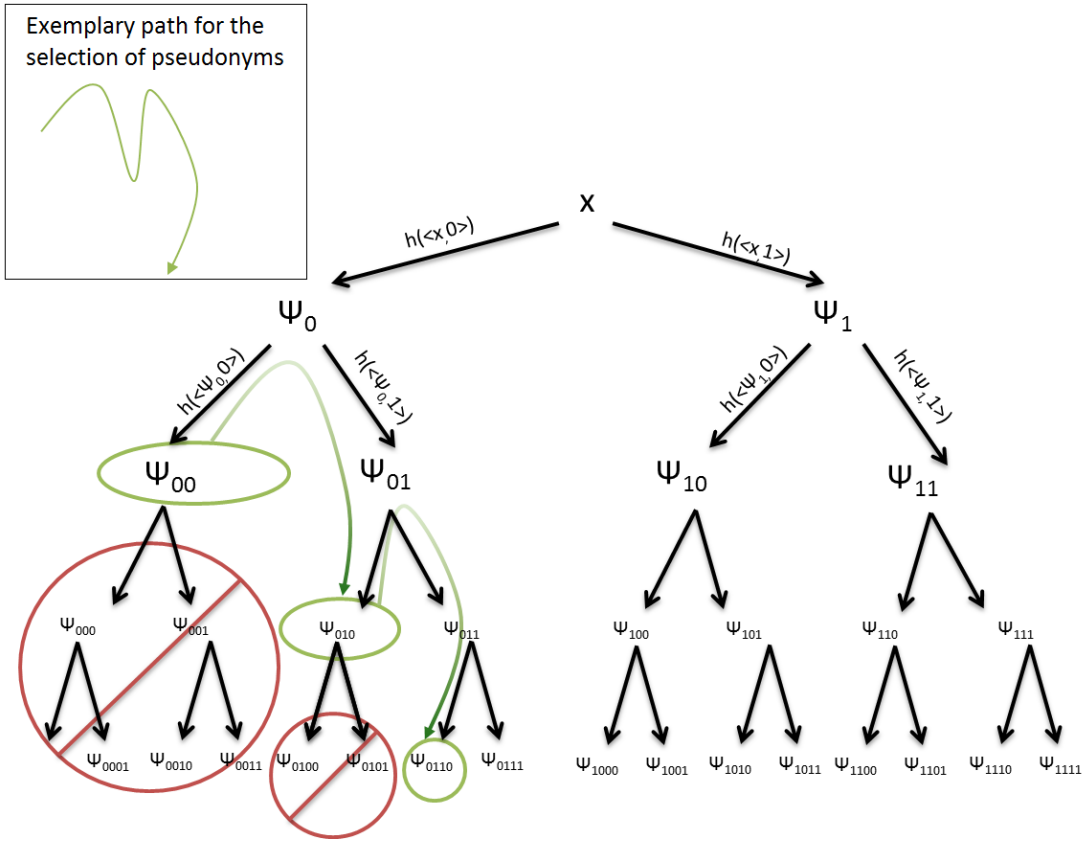

Figure 4.22: Selection of adequate outputs as pseudonyms

The value generation in Figure 4.22 follows four steps.

Step 1 – The first, initial value x is used to generate the first two levels of the tree. The first usable pseudonyms are those in the second level, generated by concatenating zeroes, namely ψ_{00} and ψ_{10} (the path of ψ_{10} is not sketched in Figure 4.22.). An entity "A" could now identify itself as ψ_{00} towards a second entity and again as ψ_{10} towards a third entity, instead of using "A". These values may not be used to generate further pseudonyms, that

means, that the potential branches beneath them may not be calculated, see the crossed areas in Figure 4.22. For the next round of secrets, the parties prepare to "jump" leaves:

Step 2 – The next input will be the sibling leaf of the last used pseudonym. Assume that ψ_{00} was the last pseudonym, which means that ψ_{01} will be used to generate the next round of outputs. ψ_{01} is now concatenated again with "0" and "1", respectively. The hash-function computes now two new values, namely the leaves ψ_{010} and ψ_{011}. The output which was generated by concatenating a zero as the new pseudonym is again used, namely ψ_{010}.

Steps 1 and 2 repeat every time a pseudonym changes. These procedures will be called the *canonical jump*.

4.3.2.5 Definition of Path and Jump

A path is a Bit-string that describes how branches from a hash-tree were (or how they should be) created. Paths generate downward branches by creating descendants of a certain starting leaf. For example, a path 00010 denotes that a hash-tree is generated by following the description of Section 4.3.2.3 until the leaf ψ_{00010} is reached.

A jump is a form of path, which combines a Bit-string with moving directions. A jump firstly moves up from its current leaf (one or several leaves) and then generates or traverses a different branch downwards. The canonical jump for example moves one leaf up, generates the opposing leaf and its left descendant.

4.3.2.6 Optimization

The canonical jump is just a suggestion to help in choosing adequate leafs as pseudonyms. Another suggestion is the dynamical generation of branches: The hash-tree is not generated entirely, but every branch is generated on demand, after a pseudonym was used. This could be done by saving three variables, the root value x (permanently), the next parent value ψ_j and the current pseudonym ψ_i, where ψ_j and ψ_i are sibling values.

Algorithm 8 shows one possible formalization.

Algorithm 8 Creating pseudonym data structure

1: Start with setting value for root node: $x \leftarrow rootsecret$
2: **procedure** GENERATENEWPESUDONYM(x)
3: $\psi_0 \leftarrow h(< x, 0 >)$
4: $\psi_{00} \leftarrow h(< \psi_0, 0 >)$
5: $\psi_{01} \leftarrow h(< \psi_0, 1 >)$
6: $x' \leftarrow \psi_{01}$ ▷ Set new parent value.
7: Return ψ_{00} ▷ Return pseudonym.
8: **end procedure**

4.3.2.7 Changing Pseudonyms

Generating new pseudonyms is done with the canonical jump. The mechanism is based on hash-functions which are easy and fast computational mechanisms and therefore well suited for constrained IoT devices. The question in focus when discussing changing pseudonyms is when to generate new ones. Pseudonym exchange has been heavily surveyed in vehicular ad-hoc networks, but the results can be transferred to any other system using pseudonyms. The reader is reminded of the related work detailed in Section 4.3.2.1: important secure pseudonym exchanging concepts can be categorized into spatial concepts, time-related concepts and user-oriented concepts. Spatial concepts are best represented in mix-zones [BS04], where pseudonyms are exchanged when system participants meet physically, although virtual mix-zones for an artificial pseudonym change have been proposed [MKW08]. Time-related mechanisms propose to change pseudonyms after a certain time, where a secure pseudonym exchange is only possible when the changing participant is not participating in the system any more. One possible solution is a so called silent period [HMYS05]. This means that a system participant stops his participation for a short time until his pseudonym is changed successfully. User-oriented concepts allow the user to decide when he wants to change his current identity. The decision can hereby be completely subjective, allowing to define own policies and thresholds for the pseudonym change. Such concepts are Swing & Swap [LSHP06] and SLOW [BHWW09]. Although all of these concepts refer to location based systems, they can be used in IoT scenarios, e.g., where pseudonyms expire and trigger a silent period for data collection or where wearable medical devices form a mix-zone and call for a pseudonym change. The question, which of these concepts is usable, depends on

the type of IoT scenario, as a silent period, a policy based approach or a mix-zone might be or might not be possible.

4.3.2.8 Pseudonym Agreement

Occasionally, a pseudonym has to be re-linked to a system participant for some scenarios. For example when a user wants to access one of his devices and the device's identity is pseudonymised or, in case of billing a service, or for the liability of a user in case of damage. The proposed mechanism for pseudonym agreement is dynamical, the parties that agree on pseudonyms do not share a list of identities or pseudonyms. They generate the pseudonyms that either one or both entities are going to use, without knowing their real identities. This is possible, if the two entities share a common root secret as described in 4.3.2.4. Note that the shared root secret does not necessarily have to correspond with the root secret of the entity that originally protected itself with pseudonyms.

New pseudonyms are generated depending on time (see Section 4.3.2.7) and method (see Section 4.3.2.4). To demonstrate the pseudonym agreement mechanism, the following is assumed: a new pseudonym is generated when a predefined timeslot expires and it is generated with the identity of one of the entities. In the following example the agreement mechanism is based on top-down hash-trees:

A user's device sends consumption data to a cloud provider. The device protects its identity with pseudonyms generated with top-down hash-trees. The user wants to know his consumption data and asks the service provider for the data of his device. He has to generate the pseudonym that the device has used to retrieve his data.

Figure 4.23 depicts how both, the device and the user, generate pseudonyms to transmit and retrieve the data from the cloud provider.

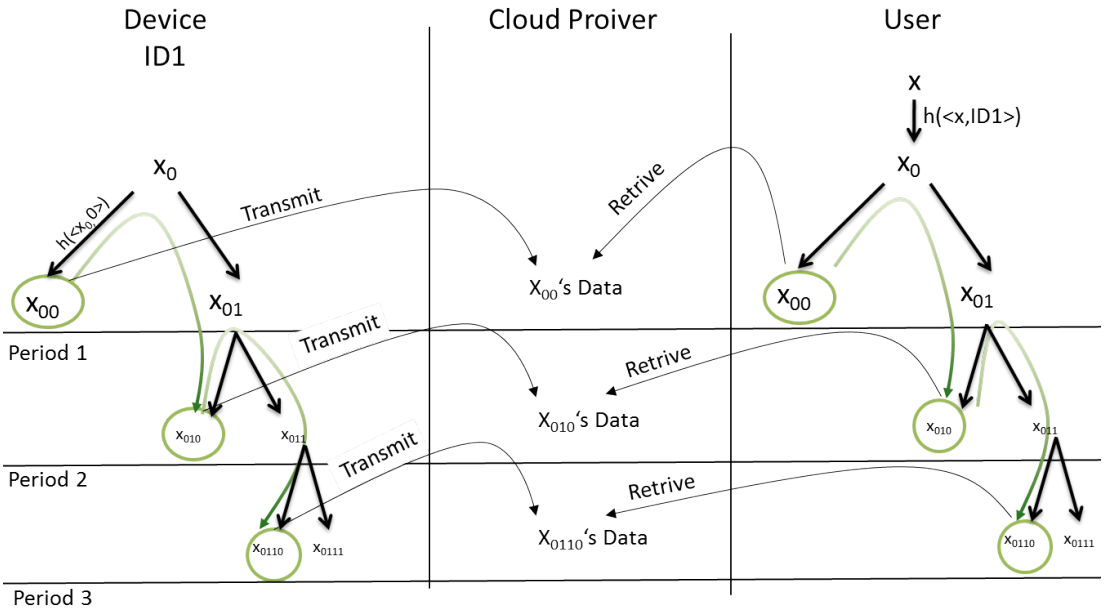

Figure 4.23: Example of a pseudonym agreement for data retrieval

The device's root secret x_0 is a sub-secret from the user generated with the user's root secret x and the device's ID. The device changes its pseudonym according predefined periods and transmits its data to the cloud provider. If the user wants to retrieve the data from the cloud provider, he computes the device's root secret x_0 and generates the pseudonym according to the period that he wants to retrieve the data from. It is assumed that the cloud provider saves all data from any pseudonym, as long as the device is able to authenticate itself as a customer of the cloud provider. Anonymous authentication mechanisms maybe group signatures [CVH91] or the PAT protocol, see Section 4.2.5.

In another example, ownership may play a role. A user could receive a data set from a pseudonym. The data set might be signed with a group signature of one of his devices, as described in the ownership mechanism of Section 2.5.3.4, such that he can be sure that the pseudonyms used were previously agreed on. The search algorithm to reconstruct the used pseudonym could be implemented as shown in Figure 4.24.

The user has a limited amount of devices for which he is able to generate pseudonyms of the according time period. He authenticates the incoming data sets and reads the period which the data set was generated. The user generates the pseudonyms of the devices that come into consideration (probably not all devices produce this kind of data).

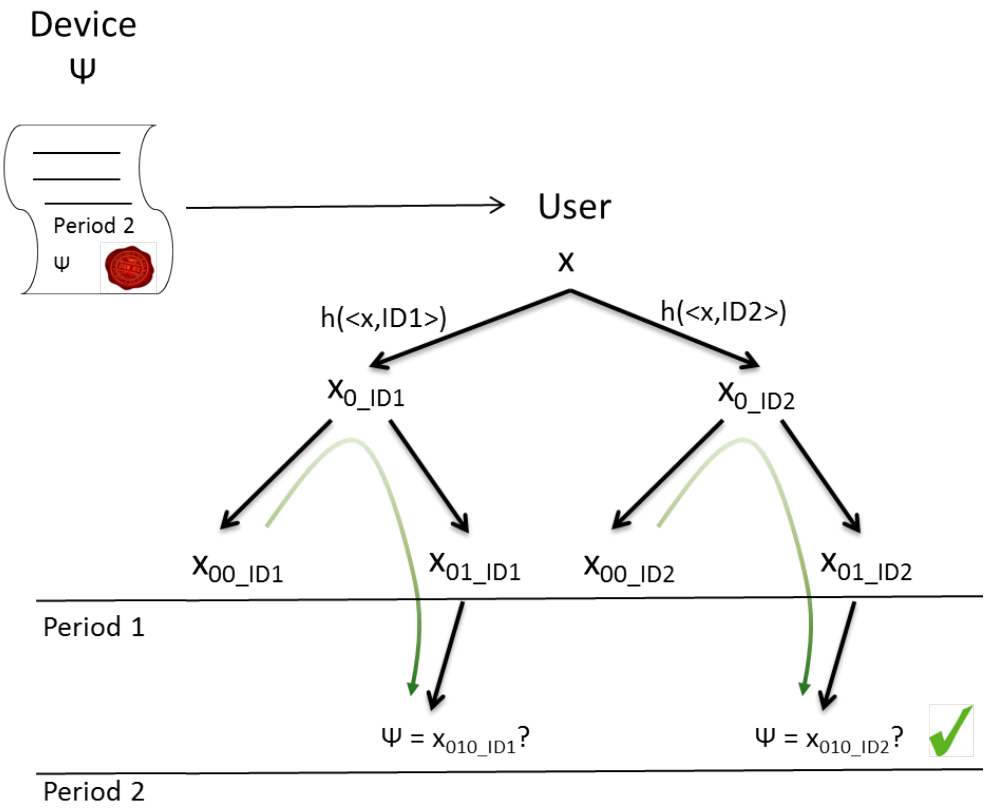

Figure 4.24: Example of re-constructing a pseudonym by a device owner

It should be noted that the computational capacity of the reconstruction is considered to be high and the computational time does not equal a full binary or n-ary tree search, as the user knows exactly which values of which branches he has to compute. In an optimized version, the user knows which periods are not needed anymore and he can start generating pseudonyms via a local path.

4.3.2.9 Summary and Future Work

The pseudonym data structure is simple, computational and battery efficient due to the use of one-way functions (hashes). The mechanism allows dynamical access to pseudonyms for single devices, fast and secure pseudonym agreement, management and revocation.

Furthermore, the pseudonym agreement does not depend on asymmetric cryptography or extensive pseudonym-to-ID tables and can therefore retain the targeted efficiency.

The top-down hash-tree data structure is used in a varied form by the PAT protocol and will also be used for the malleable message authentication codes

and group message authentication code schemes. The structure itself is purely methodical and may change in an actual implementation as pointed out in Section 4.3.2.6. There are many more implementation options using native data structures of programming languages which are going to be evaluated as future work.

4.3.3 Malleable Message Authentication Codes

When sending a message over a communication medium, it is important for the recipient of the message to be able to verify if the message was originally produced by the expected party and if it has not been modified during the transmission. This well-known property is the integrity of the message: a recipient is somehow assured of the integrity of the message. But in some situations, it is necessary that an authorized intermediate party is able to modify the message, in a restricted way. Restriction means that the intermediate party should be allowed to modify some previously defined parts of the message, while not being able to modify the others.

In other words the problem to solve is the following: an entity S, the *Signer* of a document, wants to write a message with two types of content: one is modifiable (also, depending on the modification, termed in related literature as: sanitizable, admissible or malleable) and the other is fixed. The Signer sends the message to a recipient over some communication channel. An intermediary, the censor (or *Sanitizer*) party, is allowed to change the admissible part of the message but not the fixed part of the message. The recipient or *Verifier* of the message (or rather: of the integrity of the message) should be able to verify that the message was originally produced by the expected party (the Signer) and that it has not been modified during the transmission except, possibly, for changes done to admissible parts by authorized intermediate Sanitizers. This property is the relaxed integrity of the message in the context of sanitization. In cryptography, a solution for the integrity problem is the use of signatures (based on asymmetric cryptography) and message authentication codes (MACs, based on symmetric cryptography). For the relaxed integrity problem, e.g. the concept of a Sanitizable Signature Scheme (SSS), see [ACDMT05], has been proposed, which allows authorized

censors to modify the admissible parts of a signed message without interacting with the original Signer. Implementations of SSS use a specific arithmetic. For IoT, this means the arithmetic has to be implemented on a constrained device first, as no off-the-shelf solutions are available. The arithmetic has to be optimized for each specific device which might be costly or impractical.

Thus, the problem can be casted in a more particular way. A scheme has to be designed that uses Message Authentication Codes and hash functions, as they are available for constrained devices either on hardware and/or software. This scheme has to be based on symmetric cryptography ans has to resemble a malleable signature scheme, such as SSS, that allows modifying a MAC without interacting with the original Signer.

Furthermore following properties have to be supported:

If desired, the Verifier is not able to know if a message was sanitized or not. Otherwise, it should be possible for the Verifier to check that the message was not sanitized or, if in fact it was sanitized.

The Signer is able to revoke at any time the ability of a Censor or a Verifier to sanitize (or to verify) future messages.

The method should be usable in scenarios where participants have limited power and constrained computational resources, as in the Internet of Things. A cryptographic solution should rely solely on symmetric cryptography, which is more efficient for such purposes (known SSS are based on asymmetric cryptography).

4.3.3.1 Introduction to Homomorphism

The admissibility of modifications in cryptographic schemes is known as homomorphism. Homomorphism in signature schemes allows a controlled transformation of signature-message-pairs, while preserving the authenticity and integrity conditions provided in classic digital signature schemes. The transformation can be used to change parts of a message in order to distribute information to different parties with different granularity. The categorization of allowed transformations comes in three flavours:

blackening parts of information, as seen in government documents, is resembled in

sanitization schemes (see [ACDMT05]), *omitting parts* of a message is achieved with redactable signature schemes (see [JMSW02]) and the *transformation of a predefined part* of a message is obtained by malleable schemes (see [JMSW02, CKLM13]). Successively, a malleable scheme can resemble sanitization and redaction. For simplicity, all schemes are summarized here as homomorphic and authenticator signature schemes, a detailed view on related work is given in Section 4.3.3.2. Many potential scenarios have been proposed for homomorphic schemes before. Ateniese et al. [ACDMT05] referred to multicast and database applications, medical applications, secure routing and sanitization of classified government documents to underline the scheme's utility. Other areas of potential use have been mentioned in [CKLM13] and [PPS+13] such as anonymous credential systems and privacy enhancement in constrained devices. Malleable Message Authentication Codes (MallMACs), the scheme presented in this Section[31], targets specifically the latter. The setup is explained in Section 4.3.3.3. Privacy enhancing technologies in the context of the Internet of Things have been steadily discussed (see e.g. [May09b, Web10b, AIM10b]). The focus of the discussions have been on privacy technologies that are suited for constrained devices, pointing at schemes such as [BBS04] and [PPS+13]. Though these schemes may be applied in IoT, three major constraints are regarded as the main reasons of why privacy enhancing technologies have not been popular in IoT products and industries overall:

Arithmetic for different schemes needed in small devices (e.g. chameleon hashes used in sanitizable signatures, see [ACDMT05]) are often too specific and unavailable in off-the-shelf or standard security libraries (e.g. OpenSSL).

Specific implementations are costly in software development and specially in embedded systems engineering.

Asymmetric cryptography, even elliptic curve cryptography, is demanding for battery powered devices, as formulated in Section 4.2. Battery maintenance is costly and thus a decision factor for constrained devices, see [DL01].

The reader is reminded of the IETF classification of most common IoT devices: class 1, with a capacity of 10 KB RAM memory and 100 KB flash storage (see [BEK14, KSH+12]). Albeit unproblematic for some single-processor devices

[31] The content of this Section has been partially published previously as a Siemens AG intellectual property item (patent numbers 102015205111.0, 10 2015 205 111).

(such as x-berry computers), asymmetric cryptography does not perform well on IoT devices which are based on hardware that is of low capacity.

To gain more insight in applicable schemes for constrained devices, related work is reviewed in the next Subsection 4.3.3.2. Here, state of the art malleable signatures are discussed.

4.3.3.2 Related Work

In this Section, related work is reviewed and efforts in homomorphic signatures and message authentication codes are classified according to different functionality.

Homomorphic Signatures. Johnson et al. refer to Rivest in [JMSW02] as the coiner of the term "homomorphic signatures". Rivest describes signature schemes with it, that can be partially modified. Micali et al. introduced a first set of homomorphic signatures with transitive signatures in [MR02].

In transitive signatures, two signatures of an edge x,y and y,z can produce the proof that a third signature of x,z is on the same edge. The scheme does not allow explicit modification of a message part which is required for the privacy protection targeted in MallMACs. The scheme uses RSA group homomorphism, which can be seldomly used by constrained devices due to its large key size and heavy computation, see [WGE+05].

Johnson et al. introduced in [JMSW02] redactable signatures to allow explicit deletion of sub-strings in a signed message. The scheme signs a message x in three steps: first, the message is divided into n parts, say from x_0 to x_n. In step 2 a GGM tree, see [GGM86], is generated until n leaves are created and n leaf values can be associated to each message part. The leaf values k_1 to k_n are created from a key k_ϵ. Step 3 is the creation of a Merkle-tree (see [Mer90]) over every message part x_i and according value k_i. The top-hash is signed once, put together with the key k_ϵ and re-signed to create the signature of the message x. Redacting a message means deleting a message part x_i as well as its key value k_i and generating a new top-hash including its signature. Due to the Merkle-tree, there is no need to recreate ascendant hashes, thus the top-hash is regenerated from the top-most tree ascendant. This is used to make the scheme faster and the signatures shorter.

The scheme breaks the message into smaller parts, this method is still used as a basic mechanism in recent homomorphic signature proposals. MallMACs also break the message into sub-parts and use hash-trees for secret management, which make the schemes similar, albeit avoiding RSA accumulators which are used for the signature homomorphism in Johnson et al.'s proposal.

Steinfeld et al. presented "content extraction signatures" in [SBZ02], where a part of a signed message could be extracted and still be verified with the original signature. The scheme is very similar to [JMSW02], using Merkle-trees and pseudo-random numbers replacing the GGM tree values.

Ateniese et al. have introduced the notion of sanitizable signatures in [ACDMT05]. As a reminder, a sanitized message is a message with unreadable or "blackened" parts. In [ACDMT05] this is achieved by utilizing chameleon hash-functions (see [KR00]). The scheme works as follows: A message x is again broken into n parts. The Signer of the message decides which parts may be sanitized and which parts may not. For the sanitizable parts $x_s = x_i$ to x_m with $m < n$, a chameleon hash function $f()$ generates a chameleon hash CH of the form $fx_i, x_{i+1}, ...x_{msksign} = CH$ utilizing a key $sksign$ of a RSA key pair. The rest of the message $x_fix = xwithoutx_i, ..., x_m$ is hashed with any one-way hash function to obtain H. For simplicity, it is assumed that H and CH are concatenated to $H||CH$ and then signed with a RSA digital signature scheme. The sanitization is done with the corresponding counterpart of the key pair $pksanit$, where the *collision function* of the chameleon hash finds a Bit-string CH' that is the exact collision of CH. Thus the RSA signature of $H||CH$ equals the signature of $H||CH'$. MallMACs and sanitizable signatures share the chopping of the message in sub-parts and the definition of a sanitizable (or malleable) set of the message's elements. Chameleon hashes are based on prime order cyclic groups with homomorphic properties, similar to those of RSA. For MallMACs in IoT, any mechanism that uses the factoring problem is avoided due to a possibly high battery and computational power consumption (for details, see [WGE+05]). Additionally, chameleon functions are rarely found in common cryptographic libraries such as OpenSSL, Libcrypto [CMV02] or Relic [OAG+11].

In [PPS+13] Poehls et al. show that homomorphic signatures can be imple-

mented in a constrained environment. The effort displays an implementation of state-of-the-art redactable and sanitizable schemes on a Java Card V2.2.1. Java Cards resemble constrained devices targeted by MallMACs with 10 KB of RAM and 100 KB of Flash memory (even less for the v2.2.x series, see [Mic02]). MallMACs target battery efficiency, which is only a constraint in contact less smart cards. Although other constraints exist which may also play a role, such as code size. The reader is reminded of Section 4.2 and why signature based schemes are unfavourable for battery powered devices, even if implementations exist and computational overhead is tolerable.

Homomorphic Message Authentication Codes. A more recent approach to homomorphism in symmetrical authentication is presented in [AB09, CJ11] to prevent pollution attacks in network coding. Pollution occurs when network nodes are allowed to rewrite information in packages (this is called network coding) and malicious nodes intentionally falsify or flood the network with undeliverable packages. Linear network coding algorithms break a message into n vectors of a linear space, see [KM03]. Homomorphic MACs therefore operate in linear vector spaces [AB09] or finite fields [CJ11] to achieve the desired homomorphism. MallMACs are designed for privacy scenarios and therefore are not limited to scenarios where data structures are only vectors. This allows the application of MallMACs on unprocessed Bit-strings by using very simplistic one-way functions.

In [GW13], Gennaro et al. propose fully homomorphic message authenticators, which verify the authenticity of an altered message y as an output of a process ρm, where m is the original message. If a message is sanitized or changed for personal reasons, it might be undesirable to let the Verifier know which process altered the original message, making the proposal of Gennaro et al. unsuitable for all privacy scenarios.

4.3.3.3 Setup

Section 4.3.3.1 listed scenarios where homomorphic schemes can be useful. In this Section, the scenarios are abstracted in order to provide a general setup for homomorphic signature schemes. It should be noted that SSS is used here,

but it is only the specification of a particular functionality; there are many ways to implement an SSS. The scheme used here follows the notation proposed in [ACDMT05]. Also note that the proposed MallMAC scheme is not bound to this specific SSS in particular, because it uses asymmetric cryptography as its core concept. According to [ACDMT05], a basic SSS is computed the following way:

Signing the message. The first party, called Signer, wants to sign a message M with his private key $prvK_{SIG}$ of a private/public key-pair, where he defines which parts may be sanitized by another party called Sanitizer. In any case, a third party called the Verifier should be able to verify that the message is genuine, that it was signed by the Signer and occasionally sanitized and resigned by the Sanitizer. The Signer decides which part of the message maybe sanitized. He splits M in a fixed message part (m_{fix}) and a sanitizable (or admissible) message part (m_{adm}).

$M = m_{fix} + m_{adm}$

Note: the + operator is used here to denote a form of concatenation: when two or more messages are concatenated by a +, the whole message contains both parts plus the information of where those parts "begin and end". Thus, from $a + b$, the two parts a and b can each be unequivocally identified by a parser. The signature functions used below use two input parameters: the key used for signing and the message that is to be signed. Thus $Sig_x = sign(k, m)$ denotes a signature x with key k over a message m. The Signer signs the fixed part of the message and concatenates the public key of the Sanitizer $pubK_{SAN}$, to allow the Verifier to check the signature in case the Sanitizer changes the message,

$Sign_1 = sign\left(prvK_{SIG}, m_{fix} + pubK_{SAN}\right)$.

He also attaches his signature over the whole message M, and sends this to the Sanitizer,

$Sign_2 = sign\left(prvK_{SIG}, m_{fix} + m_{adm}\right)$.

Signer $\rightarrow$ Sanitizer: $m_{fix} + m_{adm} + Sign_1 + Sign_2$

Sanitizing the message. The Sanitizer receives M, $Sign_1$ and $Sign2$. He checks the message and the signatures. If the Sanitizer wants to change the message, then the Sanitizer changes the admissible part from the message

and creates a new one (m'_{adm}). He then removes the original $Sign_2$ from the Signer and attaches a new signature $Sign'_2$ for the whole message with his private key $prvK_{SAN}$ of his private/public key-pair. The corresponding public key of the Sanitizer $pubK_{SAN}$ was originally signed by the Signer in $Sign_1$.

$m_{fix} + m'_{adm} + Sign_1 \leftarrow sign\left(prvK_{SIG}, m_{fix} + pubK_{SAN}\right) + Sign'_2$

$\leftarrow sign\left(prvK_{SAN}, m_{fix} + m'_{adm}\right) + pubK_{SAN}$

Sanitizer $\rightarrow$ Verifier: $m_{fix} + m'_{adm} + Sign_1 + Sign'_2 + pubK_{SAN}$

Verfiying the message. The Verifier knows the public key of the Signer, receives the new message $M = m_{fix} + m'_{adm}$, the public key of the Sanitizer, the signature for the fixed part by the Signer $Sign_1 \leftarrow sign\left(prvK_{SIG}, m_{fix} + pubK_{SAN}\right)$ and with it, the verification that sanitization was occasionally done by a trusted Sanitizer. The Verifier also receives $Sign'_2$; the signature for the whole message from the Sanitizer. The Verifier can now check both signatures with the public keys of the Signer and Sanitizer, and verify that the message is authentic and integer.

4.3.3.4 MallMAC Scheme - Fundamentals

Our proposed sanitization scheme is based on Message Authentication Codes for ensuring integrity and authenticity of a message. The key management for the Signer, the censor party (or Sanitizer) and the Verifier are based on hash-trees as described in Section 4.3.2.2, only that this time the hash values are used as secrets. For the reader's convenience, this Section reviews these concepts and techniques briefly and informally.

Hash-functions. Following sets exist: a set of binary sequences of length n by $\{0,1\}^n$, the set of all sequences of n - $0s$ or $1s$, by $\{0,1\}^*$, the set of binary sequences of any length, including the empty sequence ϵ of length 0, and by $\{0,1\}^{n+}$, the set of finite binary sequences of at least length n. A hash-function $h()$ is a function of $\{0,1\}^{n+}$ to $0,1^{n+}$, which takes binary sequences of a minimum length n and produces binary sequences of length n, where following properties apply:

One-way property or non-invertible property: given the knowledge of $h()$ and

y, it is computationally impossible to compute x with $h(x) = y$. *Collision resistance:* it is very unlikely to find two (or more) values x_1, x_2, where $h(x_1) = y$ and $h(x_2) = y$. *Chaos:* even similar values generate significantly different outputs: changing a value by even one Bit should generate an output that is about 50% different than the output of the unchanged value.

Message Authentication Codes. Recapitulated from [KBC97]: A message authentication code is an output generated from a message and a secret to provide integrity and authenticity assurances on the message. Integrity assurances detect accidental and intentional message changes, while authenticity assurances affirm the message's origin. HMACs are only one method to generate MACs; the specific MAC generation method is irrelevant for the rest of the approach. A keyed-hash message authentication code (HMAC) is a specific construction for calculating a message authentication code (MAC) involving a hash function in combination with a secret key. As with any MAC, it may be used to simultaneously verify both, the data integrity and the authentication of a message.

Virtually unbounded amount of secrets based on hash-trees. The use of hash-trees, as described in Section 4.3.2.2, is a top-down approach that allows the generation of an unbounded amount of pseudo-random numbers based on one seed which should be kept as a secret. The numbers are used as secrets for encryption or MAC generation. It also allows the semi-trusted Sanitizer to know a limited part of the shared symmetric secrets, with the addition, that the secrets of the Sanitizer and the Verifier can be revoked. By using hash-functions for both, the hash-tree and signature generation, implementation complexity is simplified. The reason for using a hash-tree as a design decision is that binary trees are well-known. The adjective binary refers to the number of children for any node, not to the values of the nodes. Trees can be seen as a particular type of acyclic directed graphs or as data structures used to index some values (it is assumed that those values are integers). A binary tree can be seen as a set of nodes, each node being a pair (i, x_i), the address of the node (a sequence of $0s$ and $1s$) and the value of the node (an integer), such that for each address i there is only one value

x_i. A binary tree can be defined by induction as follows: start with a node, the root of the tree, and assign a value x to this node. Then, construct two further nodes, the left child of the root and the right child of the root and assign values to them, x_0 and x_1 and joint the root with those two children by a directed edge. This procedure is repeated indefinitely: create a left and a right child for each node and assign values to them, etc. Each node has – in this way – two children, one left and one right. And each node has one parent, except for the root which has no parents. Each node is associated to a value (or each node has a certain value).

A node A is a descendant of another node B if A is a child of B or (by induction) the child of a node that is a descendant of B. In that case, B is called an ancestor of A. The address of a node is defined as a finite sequence of zeros and ones, inductively defined as follows: the root has address ϵ (the empty sequence). If a node has the address i, then the left and right children have addresses i_0 and i_1, respectively. Thus the address of a node is an element of $\{0,1\}^*$ (the set of binary sequences of any length) that simply describes the sequence of steps to reach the node starting from the root: a "0" means "left" a "1" means "right". The terminology x_i for the value of the node at address i. The address of a node will be also called the global path of the node. A (local) path is a Bit-string that describes how to reach a certain node starting from another one. Thus, from a given node, paths lead to the descendants of that starting node. From an implementation point of view, traversing a tree is moving in a sequence of steps from one node to one of the node's children or the parent. Traversing a tree implies being able to read the values x_i, of the nodes reached. The top-down hash trees are constructed using hashes: the children of any node are constructed using two hashes of the node. Thus, starting from any point in the graph, it is easy to move downwards along any path, it requires only calculating some hashes. But moving up in a tree is not possible without extra knowledge, as this would require inverting a hash. The jump is closely related to a path, but the movement is not only moving downwards: a jump first moves up one or several times from its current node and then moves down along a

path. A jump can be represented as a sequence of one or more "-" (minus signs) followed by a sequence of $0s$ and $1s$.

The reader is reminded of the *canonical jump*. It moves one node up, moves to the right child and from there to its left child. The canonical jump is given by the sequence -10 ("up, then right, then left"). The top-down hash-trees are constructed using a parameter n (an integer) an initial seed (or "root secret") represented as a binary sequence of length n and a hash-function with range in $\{0,1\}^n$. The construction goes as follows: The initial seed x is the value associated to the root (notice that the "subindex" for x is the address of the root: ϵ, the empty sequence). If the node at address i has value x_i, then the value of the left child is $x_{i0} = h(< x_i, 0 >)$ and the value of the right child is $x_{i1} = h(< x_i, 1 >)$ where $<,>$ denotes concatenation. Notice that due to the chaos and the one-way properties of the hash, the values of x_{i0} and x_{i1} are pseudo-random and there is no efficient algorithm to construct one from the other without knowing the common parent.

In MallMACs the values on the tree are used as private secrets, shared secrets or one-time secrets. A private secret is a secret that an entity shares with no one, but it is used for challenge-response protocols, to create further secrets (which may be shared), or to demonstrate to a revision or audit authority that some values are related to this secret. A shared secret is a secret that several (usually a small number of) entities know, but it is not intended to be disclosed in public. A one-time secret is a secret that is used only once to prove that the originator knows a particular shared secret. Such one-time secrets can be directly or indirectly disclosed publicly when used or after being used. This is the reason why they are used only once, as soon as they are disclosed or leaked, they are not interesting as secrets any more.

4.3.3.5 The MallMAC Scheme

In this Section the proposed scheme for *Malleable Message Authentication Codes* is presented. The setup of the MallMAC scheme resembles that of Sanitizable Signatures [ACDMT05] with three entities: the Signer, the Verifier and the

Sanitizer. All entities share some secrets which correspond to certain positions of one binary graph. The Signer has the root secret x. He is able to generate every node of the hash-tree (or: he "knows" the whole tree). The Verifier's initial secret is a node in the tree and therefore a descendant of the hash-tree's root secret (e.g., x_0 in Figure 4.25). Thus the Verifier is able to verify some MACs created by the Signer, but not all, and is not be able to forge MACs with keys that are not descendants of the Verifier's initial secret. The Sanitizer knows Sz, a secret descendant of the initial secret of the Verifier. This implies that the Signer and the Verifier are able to generate many common secrets and in particular all the "sanitization secrets" of the Sanitizer.

Additionally, the Signer and Sanitizer share a common secret to allow the Sanitizer to verify the incoming message from the Signer. This secret S is a node in the tree that is not in the path to the Verifier's initial secret (x0 in Figure 4.25).

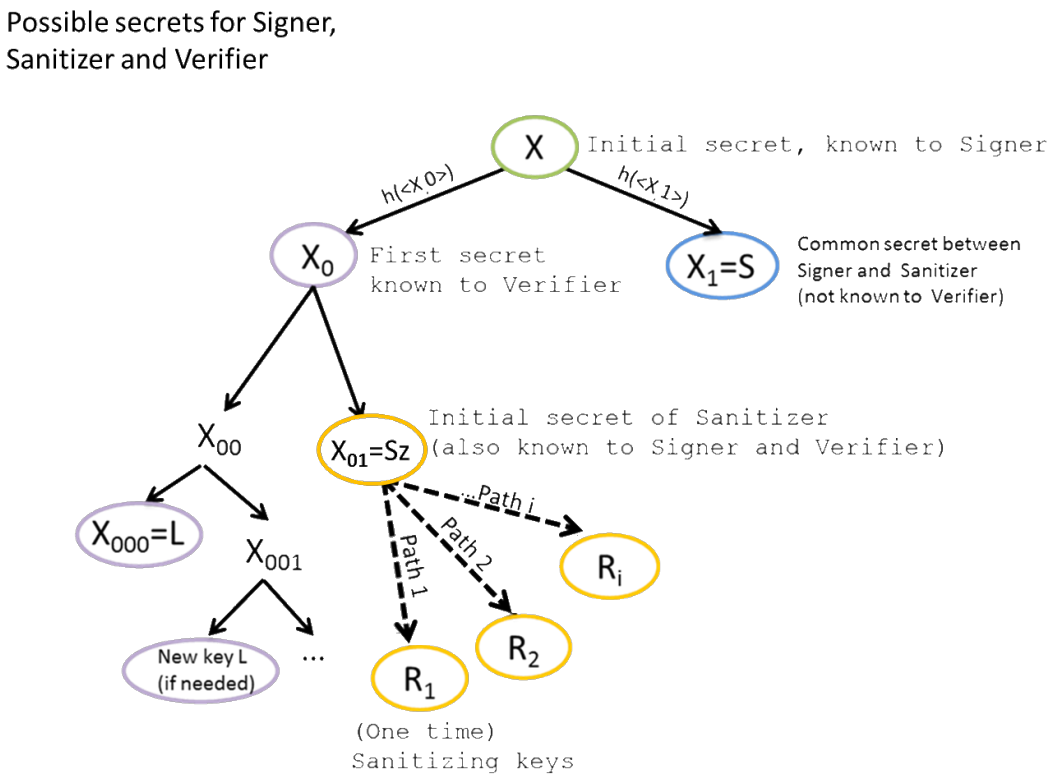

Figure 4.25: Possible secrets and paths for Signer, Sanitizer and Verifier

Following the initial setup, the scheme defines a three step protocol. Figure 4.26 sketches the protocol's data flow in a sequence diagram. Each step corresponds to

the actions that the Signer, Sanitizer and Verifier are allowed to do. The actions are:

Steps of the Signer. The Signer has a message, which he wants to send to the Verifier. The message m is composed of two parts:

- f, contains the fixed part of the message, and a path (as described in Section 4.3.3.4) $path\_to\_R$, which allows the generation of the sanitizing key R.
- a, which solely contains the admissible part of the message.

The Signer now generates a MAC $|f|L$ (resembles Sig_1 in the basic SSS). The used secret L is the first secret of the path known to Signer and Verifier (e.g., x_{000}, see Figure 4.25). $Path\_to\_R$ is later used by the Verifier (and occasionally the Sanitizer) to generate the sanitizing key R. There are several ways to generate different sanitization keys from Sz:

- $Path\_to\_R$ can be a sequence of Bits to generate a branch with certain descendants from Sz. This is recommended if the Sanitizer's initial key needs to be replaced or refreshed.
- Instead of generating a path for every message, which might result in longer paths and higher computational costs due to the execution of hashes to generate the tree's branches, the Signer could include key material KM_i for every new sanitization key R_i, where $R_i \leftarrow h\,(< Sanitizer's\_initial\_key, KM_i >)$.

The Signer then computes a second MAC for the whole message $|m|R$ (resembles Sig_2 in the basic SSS) with the current sanitizing key R.

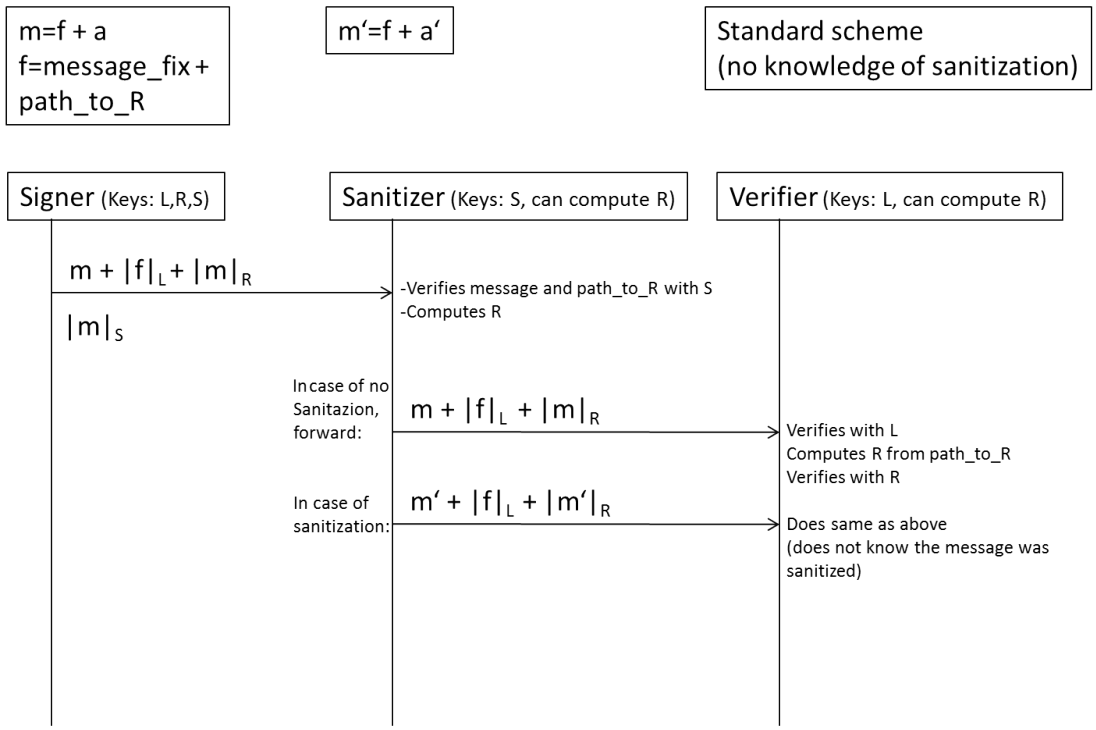

Figure 4.26: The MallMAC Protocol

An additional signature $|m|S$ is proposed, which is generated by the Signer with a special key S and is only known to Signer and Sanitizer. The purpose of this additional signature is to allow the Sanitizer verifying the message part a (the one he is allowed to sanitize) and the used *path* (to generate the sanitization key R). Finally, the Signer sends m, $|f|L$, $|m|R$, and $|m|S$ to the Sanitizer.

Steps of the Sanitizer. The Sanitizer first verifies $|m|S$ with the key S. If the verification holds, the Sanitizer is able to identify the admissible part a of the message and is able to generate the sanitization key R with *path_to_R*, where *path_to_R* is a part of the fixed message f.

Case 1 - No sanitization. If the Sanitizer agrees to the content of a, he redirects the original message and MACs from the Signer to the Verifier (m, $|f|L$, $|m|R$).

Case 2 - Sanitation of the message. The Sanitizer decides to change the admissible part of the message a, resulting in a'. He then creates a new MAC $\leftarrow |m'|R$ with $m' \leftarrow f + a'$, and the secret R. The Sanitizer now sends $m' \leftarrow f + a'$, $|f|L$, $|m'|R$ to the Verifier.

Steps of the Verifier. The Verifier knows L and is able to compute R from *path_to_R*. Two cases apply for the Verifier:

Case 1 - No sanitization. If the Verifier is able to verify $|f|L$ $|m|R$ with R, the message was left as originally signed by the Signer. The Verifier does not need to verify $|f|L$. The Verifier verifies $|f|L$ with L and computes R from the given *path* (part of f) and then verifies $|m|R$. If the verification holds, the Verifier knows the message is authentic.

Case 2 - Message was sanitized. The Verifier does the same as above, with the small difference that he verifies $|m'|R$ (and not $|m|R$); As the Verifier does not know the original message m, he is not able to recognize whether the message was sanitized or not.

Suggestions for revoking the Sanitizer's secret. The Sanitizer is not allowed to change messages by himself in the MallMAC scheme. He relies on the path given and authenticated by the Signer in step 1. If the Sanitizer is untrusted, there are several ways of revoking his permissions:

- The Signer neither signs nor forwards the path to use. This impedes the Sanitizer to sign a valid MAC for the new message.
- The Signer sends a revocation message to the Verifier, which tells him to block/disregard any breach generated by the secret Sz.

Suggestions for revoking the Verifier's secret. In case the secrets of the Verifier have to be revoked, the following ways are proposed to revoke his secret:

- The Signer changes L and publishes this in a secure way to all Verifiers that are still allowed to participate. The excluded Verifier is not be able to verify messages with the old secret.
- Additionally, the Signer changes Sz and communicates this to all Verifiers and Sanitizers, that are still allowed to participate. The excluded Verifier is not be able to generate valid sanitization keys anymore.
- The new L and R are computed from a new branch, as all descendants from the secrets of the untrusted Verifier must be avoided.

4.3.3.6 Variations of the Scheme

Several variations are possible to further enhance the revocation mechanics and the knowledge of sanitization by the Verifier.

In a first variation, the Verifier is allowed to know when a message was sanitized. Two different ways maybe used to notify the Verifier. The first option is knowledge of sanitization by different use of secrets. The proposed variation of the scheme is as shown in Figure 4.27:

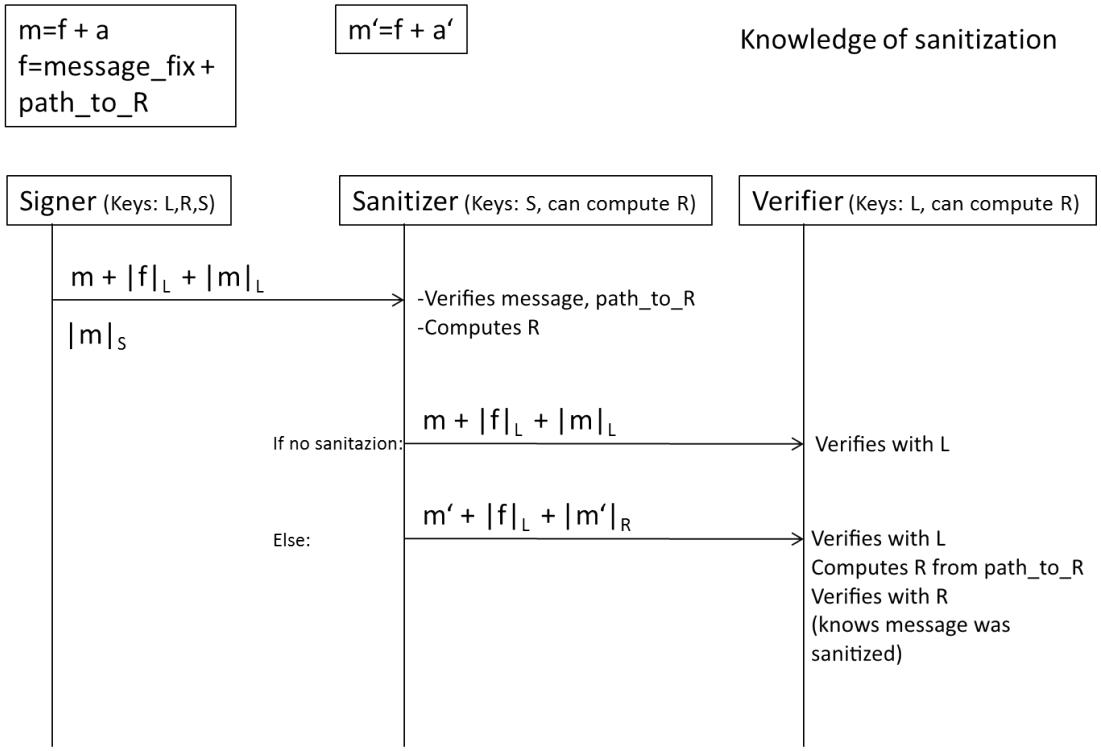

Figure 4.27: Proposed variation for sanitization knowledge by using different secrets

Steps of the Signer. Analogous to the normal scheme presented in the previous Sections, the Signer has a message, which he wants to send to the Verifier. The message m is composed as seen in Figure 4.26. The Signer sends, as opposed to the normal scheme, m, $|f|L$, $|m|L$, and $|m|S$ to the Sanitizer. That means, the message itself is verified by a MAC that is generated with the secret L (instead of R).

Steps of the Sanitizer. The Sanitizer first verifies $|m|S$ and generates R as seen in Figure 4.26.

Case 1 - No sanitization. If the Sanitizer agrees to the content of a, he redirects the original message and MACs from the Signer to the Verifier (m, $|f|L$, $|m|L$).

Case 2 - Sanitization. The Sanitizer decides to change the admissible part of the message a, resulting in a'. He then creates a new MAC $|m'|R$ with $m' \leftarrow f + a'$, and the secret R. The Sanitizer now sends $m' \leftarrow f + a'$, $|f|L$, $|m'|R$ to the Verifier.

Steps of the Verifier. The Verifier knows L:

Case 1 - No sanitization. If the Verifier is able to verify $|m|L$ with L, the message was left as originally signed by the Signer. Also, the Verifier does not need to verify $|f|L$.

Case 2 - Sanitization. The verification of $|m'|R$ with L fails. In this case, the Verifier verifies $|f|L$ with L and computes R from *path_to_R* (part of f) and retries the verification. If the verification holds, the Verifier knows the message was sanitized by a trusted party and thus remains authentic.

In this case, the Sanitizer may leave the message unchanged and still make use of the secret R. By doing this, he can make the Verifier believe that the message was changed, even if it was not. Otherwise, the Sanitizer is not able to change the message without the Verifier's notification. The reason is for this is the usage of the secret L, which is unknown to the Sanitizer, but necessary for verifying the case of no sanitization.

The second option is the knowledge of sanitization by value. The foundation is the standard scheme of Figure 4.26, but in case of sanitization, the Sanitizer adds one specific Bit to acknowledge sanitization. The scheme now works as follows:

Steps of the Signer. The Signer has a message, which he wants to send to the Verifier. The message m is composed of three parts:

- f, contains the fixed part of the message, and the Bit-string called *path_to_R*, which allows generating the sanitizing key R.
- a, which solely contains the admissible part of the message.
- b is a Bit that denotes if a message was sanitized (0 for original, 1 for sanitized).

The Signer behaves as seen in Figure 4.26, but he additionally signs $b \leftarrow 0$ when signing m. The Signer sends m, $|f|L$, $|m|R$, and $|m|S$ to the Sanitizer.

Steps of the Sanitizer. The Sanitizer verifies the received messages and generates R as seen in Figure 4.26.

Case 1 - No sanitization. If the Sanitizer agrees to the content of a, he redirects the original message and MACs from the Signer to the Verifier (m, $|f|L$, $|m|R$).

Case 2 - Sanitization. The Sanitizer decides to change the admissible part of the message a, resulting in a'. He now additionally changes $b \leftarrow 0$ to $b' \leftarrow 1$ and creates a new MAC $|m'|R$ with $m' \leftarrow f + a' + b'$, and the secret R. The Sanitizer now sends $m' \leftarrow f + a' + b'$, $|f|L$, $|m'|R$ to the Verifier.

Steps of the Verifier. The Verifier acts the same as seen in Figure 4.26. He now additionally verifies b and knows from b's value, if the message was sanitized. It should be noted that in this case the Sanitizer is free to choose b. This means, that the Sanitizer may or may not change the admissible part and lie about the sanitization by changing the value of b.

Optimized secret management for better revocation. Figure 4.25 depicts one of many possible secret distributions. The proposed distribution in Figure 4.25 is not optimal in case of secret revocation, as there is no space for the Sanitizer to generate a new secret by means of the canonical jump (e.g., the Sanitizer would be using the common secret S for the new secret of the Verifier). An optimization can be achieved by including several blank nodes between each party's secret. Blank nodes allow the party with knowledge of the superseding secret to revoke another party's secrets by means of the canonical jump. Figure 4.28 displays the possibilities given by including several blank nodes before and after each of the party's initial secrets.

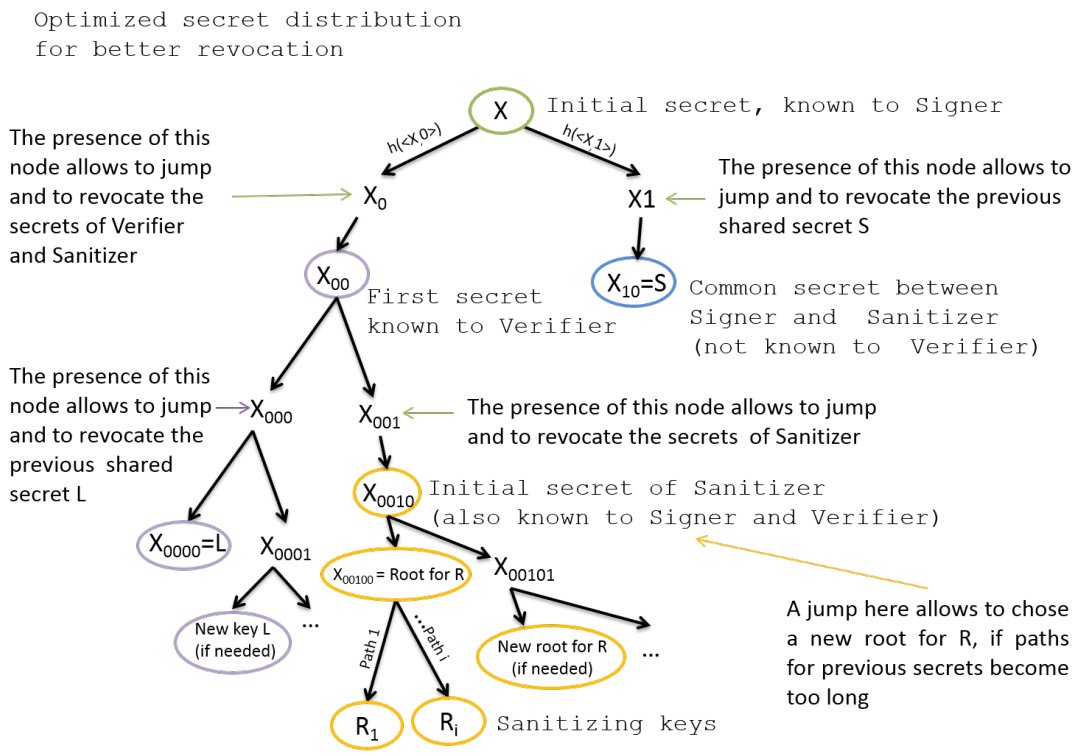

Figure 4.28: Variation of the distribution of secrets for better revocation

The colors of the description arrows indicate which party may act on a specific node. E.g., by leaving x_0 unused, the Signer may use the canonical jump (or other, more complex jumps, i.e. moving up in the tree and choosing different branches) to revoke any of the secrets known to the Verifier and the Sanitizer and to generate a new branch of secrets. In the same fashion, the Verifier or the Signer may use the canonical jump starting from x_{0010} to revoke any of the secrets known to the Sanitizer. This optimized secret distribution allows the revocation of secrets without having to generate a complete new hash-tree. Additionally, infinite revocations are possible due to the canonical jump and the nature of top-down hash-trees.

4.3.3.7 Protocol Verification

Details regarding the verification of the MallMACs protocol and the used formal and technical tools are given in Sections B.2 and B.3.

4.3.3.8 Summary and Future Work

The MallMAC scheme contributes a new malleable scheme purely based on message authentication codes. This allows using the scheme in situations where resources are restricted or where no asymmetric computation is possible. Hash-functions are already available in even the most constrained devices, thus allowing the implementation of the scheme with minimal additional effort. Also, by using hash-functions for both, MAC generation and secret generation, the scheme reduces complexity for computation and energy consumption. The modified scheme, as presented in Figure 4.28, once set up, allows infinite revocation of secrets for the Verifier and the Sanitizer. This scheme also allows variations depending on the trust relationships between all participants. Accordingly the Verifier maybe notified, if a message was sanitized.

4.3.4 Group Message Authentication Codes

Group signatures ensure unforgeability for messages, authenticate Signers and provide k-anonymity for the members of a signing group. Since the original proposal by Chaum and van Heyst [CVH91], there have been many adaptations and extensions of group signatures, for example [BBS04].

Group signatures can be used in many more scenarios to provide privacy preserving authentication. But as seen in [MFG+12], group signatures are primarily based on asymmetric cryptography, which might be to complex or cost intensive for low powered, low budget and constrained devices, such as those found in the Internet of Things. As there have been no aspirations to provide a similar mechanism with symmetric cryptography, a mechanism that resembles group signatures is proposed, entirely based on symmetric cryptography. The proposed mechanism maintains the same properties as the original proposal by Chaum and van Heyst (k-anonymity of Signers, occasional relinkability, separation of duty, revocation of secrets) and is thus applicable for the same scenarios as the asymmetric counterpart.

4.3.4.1 Introduction to Group Signatures

In this subsection group signatures are broadly described by detailing the simplified scheme of [MFG+12]. The Section lays emphasis on simplified formalisation in favour of legibility

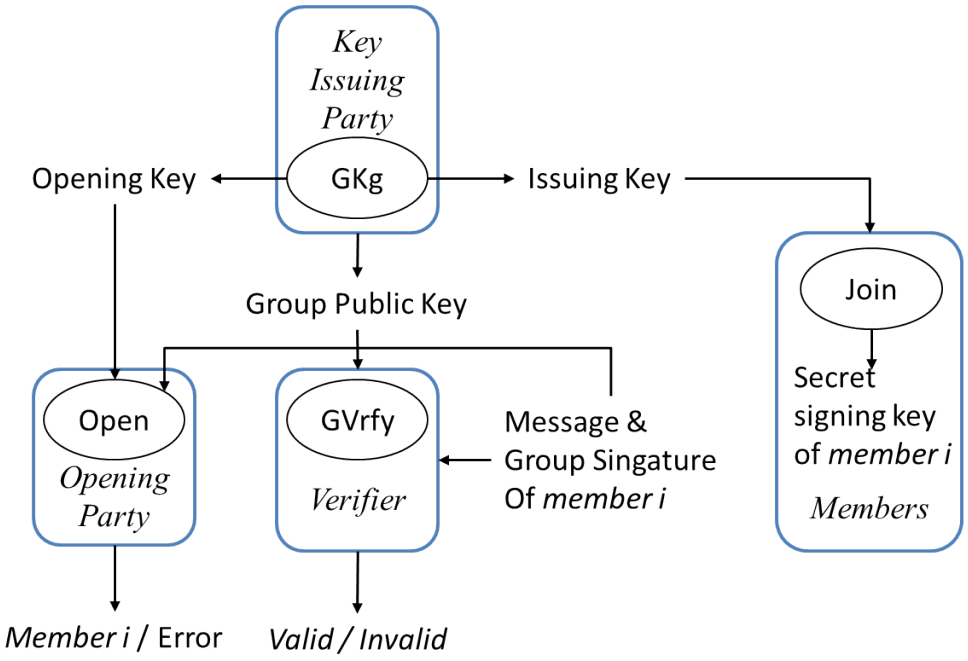

Figure 4.29: Variation of the distribution of secrets for better revocation

Figure 4.29 depicts the several parties participating in a group signature scheme. As in other privacy schemes, group signatures rely on strict separation of duties and the independent action of participants. The scheme begins with the *Key Issuing Party* which does the key management for all participants. The Issuing Party generates, distributes and revokes keys. It is not involved in the verification or opening (or "linking") procedures of the scheme. The Issuing Party is able to generate all keys with a function $GKg()$, which takes the initial size of a group as an input.

The function generates three keys as an output, the group's public key, the member keys and the opening key. Depending on the scheme, also a revocation key is generated. Algorithm 9 shows the generation steps.

Algorithm 9 Generation of keys in a Group Signature Scheme.

1: $GroupPublikKey$, $RevokeKey$, $OpeningKey$, $MemberIssuingKey_{1...j} \leftarrow GKg(number\_of\_groups)$
2: $MemberKey_{i1...in} \leftarrow GKg(initial\_members\_i, n, MemberIssuingKey_i)$

The key Issuing Party can nominate a further member Issuing Party that solely manages the generation and removal of member keys. Depending on the scenario, the Issuing Party may create several member issuing keys for different groups.

The keys are further referred to as $GPubKey$, $Memberkey_i$, $RevokeKey$ or $IssuingKey$, $GOpeningKey$. The Issuing Party then distributes the keys between the different parties. Algorithm 10 shows the distribution.

Algorithm 10 Distribution of keys in a Group Signature Scheme

1: $Member_1$ to $Member_i \leftarrow Memberkey_{1...i}$
2: Verifier(s) and Certificate Authorities $\leftarrow GPubKey$
3: Opening Party $\leftarrow GOpeningKey$

It should be noted that the Issuing Party has to remember which member received which key to assist the Opening Party in re-linking a signature to a member. This is done in a member-secret distribution table.

Additionally, the Issuing Party can add several members to the group with the function $AddMember\,()$:

Algorithm 11 Adding new members in a Group Signature Scheme

1: **procedure** ADD NEW MEMBERS($GPubKey, number\_of\_new\_members\_m$)
2: Generate $Memberkey_{i+1...i+m}$
3: $GPubKey' \leftarrow add\ Memberkey_{i+1...i+m}$
4: Return $Memberkey_{i+1...i+m}$, $GPubKey'$
5: **end procedure**

The Issuing Party can also revoke keys as detailed in Algorithm 12.

Algorithm 12 Revoking members in a Group Signature Scheme

1: **procedure** REVOKE MEMBERS($GPubKey, Memberkey_i$)
2: $GPubKey' \leftarrow remove\ Memberkey_i$
3: Return $GPubKey'$
4: **end procedure**

The public key is changed in the process. It does not longer work with the counterpart of $Memberkey_i$ which was removed from the group.

The group members join a group by obtaining secret signing keys. They are able to create digital group signatures in the name of the whole group. Neither a Verifier nor the members themselves are able to recognize the source of a signed message. A signature is created with the function $GSign\,()$:

Algorithm 13 Sign a message in a Group Signature Scheme

1: **procedure** SIGN MESSAGE($Memberkey_i, message$)
2: $Gsignature_{message} \leftarrow GSign\,(Memberkey_i, message)$
3: Return $Gsignature_{message}$
4: **end procedure**

The signature does not resemble the original message key and is verifiable with the group public key.

A Verifier has access to the group's public key and is able to verify that the group is the authentic source of a message, but he is not able to pinpoint the group member that originally signed the message. The function $GVrfy(GPubKey, Gsignaturemessage, message)$ gives the output *Valid / Invalid* in case of a valid or invalid verification of the message.

Algorithm 14 Verify a message in a Group Signature Scheme

1: **procedure** VERIFY MESSAGE($Gsignature_{message}, message, GPubKey$)
2: $Valid$ or $Invalid \leftarrow GVrfy\,(Gsignature_{message}, message, GPubKey)$
3: Return $Valid$ or $Invalid$
4: **end procedure**

The Opening Party has a unique key, which allows it to re-link a signature to one member key of the group. This property is needed in case a malicious member is culpable of fraud or has to provide compensation of damages for his misbehaviour. The Opening Party uses the function $Open\,()$ with the arguments shown in Algorithm 15.

Algorithm 15 Re-link (or "opening") a Signature in a Group Signature Scheme

1: **procedure** OPEN($GOpeningKey$, $Gsignature_{message}$, m, $key\_distribution\_table$)
2: $MemberKey_i \leftarrow Open\,(GOpeningKey, Gsignature_m, m)$
3: ▷ m is the message
4: $Member_i \leftarrow$ re-link $MemberKey_i$ with $key\_distribution\_table$
5: Return $Member_i$
6: **end procedure**

The function outputs a specific member key $Memberkey_i$. Note: the member key does not identify a user by itself. The Opening Party needs to know from the Issuing Party, which user was given the key that was generated as output from the $Open\,()$-function (exemplified in Algorithm 15 by a key distribution table). The Opening Party is supposed to have a high trust level, and only re-link signatures in justifiable circumstances. Additionally, the Opening Party should not be able to access signatures by itself. It should only have temporary access to a limited amount of signatures by means of the Verifier.

4.3.4.2 Related Work

Group signatures have been recognized as one of the major privacy enhancing technologies to reach data minimization and privacy by design [MFG+12]. Although many variations of group signatures exist, some of them intended to be used in constrained devices (e.g., see [BBS04]), they are all based on asymmetric cryptography. Albeit some of these group signatures might achieve efficient results, there are scenarios, where asymmetric mechanisms are not considered due to legacy, budget or capacity constraints for certain systems and devices. Symmetric schemes that resemble digital group signatures are few and mostly deeply nested in special cases, which are hard to transfer to other scenarios. For example, [ZLL+08] implements a privacy preserving Message Authentication Code scheme for vehicular area networks, where several cars (Signers) and a Road-Side Unit (RSU) (the Verifier) pre-share symmetric secrets. Every vehicle obtains the same pseudo-ID and one special, pre-shared key. When a vehicle generates a MAC over a message, the message is signed with the vehicle's special key. The RSU receives the message and a MAC, and it verifies the received MAC by evaluating

every pre-shared key, until it is able to confirm the authenticity of the MAC. The scheme provides k-anonymity against an outside attacker, but it does not fulfil the original idea of group signatures, where the Verifier is unable to know who the original Signer is. In the scheme proposed by [ZLL+08], the RSU will notice after the second verification which keys it has used and which vehicle is using which key to sign its MACs. Additionally, VANETs can be seen as a special scenario, where only a few vehicles are expected to be near a RSU at the same time. But if the possible Signers rise to several thousands, the pre-shared keys and the verification of one MAC with all pre-shared keys does not scale well. Finally, there is neither a mechanism for the revocation of keys, nor can this scheme be used in semi-trusted Verifier scenarios, as in the scheme of [ZLL+08], the RSU has to know every signing key and is thus able to generate MACs itself anytime. A Message Authentication Code scheme that resembles the properties of group signatures is presented here. It provides k-anonymity for Signers, separation of duty, easy key management, support of a vast amount of participants as well as a key revocation mechanism. As in the original proposal, all participants can have different trust levels; thus circumventing that one party is able to exploit the scheme by itself.

Group signature properties Following properties can be derived from the group signature scheme above.

Separation of duty. Several parties fulfil different obligations, thus circumventing mayor drawbacks, if one of them becomes untrusted (with the Issuing Party as an exception, it has to be fully trusted). Take the as an example the Opening Party; if the Opening Party becomes untrusted, there is no possibility for it to relink signatures at will, as it does not participate in the signature verification process and thus does not obtain signature message pairs, nor does it know whom the opened key belongs to. Additionally, the separation of duty is required in cases of revocation (see below) or in different use cases and variations of the scheme, were a company is able to add customers and verify group signatures (the company is the verifying party as well), but issuing all the keys is done by one government authority

(i.e., it is the Issuing Party) and relinking in cases of fraud is done by another government authority (the authority is the Opening Party).

Unlinkability. Unlinkability (also after eventual relinking) of Signers by adversaries, other Signers, Verifiers and the Issuing Party. The k-anonymity properties of the signing keys allow a Signer to remain anonymous, until the Issuing Party, the Opening Party and the Verifier work together to relink a specific signature to its original Signer. But even after relinking one or many signatures, the exposed Signer will be able to generate further signatures and remain anonymous.

Revocation of secrets. Whenever a party becomes untrusted, its secrets can be revoked without any impact on the scheme as a whole.

The Group MAC scheme will maintain the same properties, see Section 4.3.4.5.

4.3.4.3 Setup

The setup of the Group Message Authentication Codes (GroupMACs) Scheme resembles the setup of the simplified Group Signature Scheme formulated in 4.3.4.1. The participating entities are the *Key Issuing Party*, a *Certificate Authority*, one or several *Verifiers*, a group with n Members and an *Opening Party*. The duties of every party resemble those of the original as well, the Issuing Party issues keys and, if justfied, will identify the member of a group given his member key. The Opening Party links a MAC to the respective member secret. The Verifier takes public information from the CA in order to verify a given MAC. The members of a group create group message authentication codes that are (i) indistinguishable from other group message authentication codes that they have created previously or subsequently and (ii) indistinguishable from GroupMACs that other members of the group have created previously or subsequently.

4.3.4.4 The GroupMACs Scheme

The presented scheme for Message Authentication Codes with adapted k-anonymity to resemble digital group signatures is based on the same fundamentals of top-down hashes, as seen in Section 4.3.3.4. The reader is reminded of the properties of the top-down hash trees: The top-down approach allows the generation of practically

infinite secrets for encryption or MAC generation based on one secret. It also allows a semi-trusted Verifier to know a limited part of the shared symmetric secrets, including a revocation mechanism for the Verifier and untrusted members of a group. By using hash-functions for both, the hash-tree and MAC generation, implementation complexity is simplified.

Proposed scheme. Firstly, the initial secret distribution is explained. Afterwards, the Group Message Authentication Code scheme with adapted k-anonymity is detailed by explaining the every interaction of each party.

Initial secret distribution. Table 4.6 describes the initial secret distribution for each party. Note that all the secrets in Table 4.6 are pre-shared. Secrets described as "root secrets" are used to generate top-down hash-trees (see Figure 4.30).

Participant	Initial Secret
Signer	Signing root secret $MemberSecret_i$ (one per Signer); common clock.
Issuing / Opening Party	Generates a table to re-link keys and real users; generates the Verifier keys KV_i; generates root Member Secrets ($MemberSecret_1$ ... $MemberSecret_n$).
Helping Party / CA	Member Secrets ($MemberSecret_1$ ... $MemberSecret_n$); common clock; generates public verification table from Member Secrets; knows Verifier keys KV_i.
Verifier	Verifier secret KV_i (one per Verifier); requires public verification table from Helping Party.

Table 4.6: Group MACs: Initial distribution of secrets

Figure 4.30 sketches the key generation for two members, based on the root secrets that the members were given initially in the scheme.

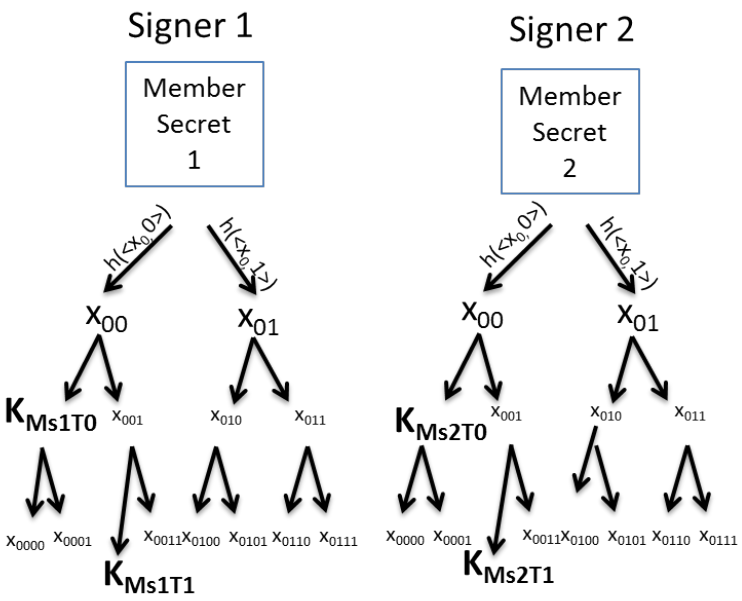

Figure 4.30: Example of existing hash trees in the scheme with two Signers

How the scheme works. The scheme is described for two members of a group, Signer 1 and Signer 2, although the scheme works for any number of Signers. One Verifier is also described (the scheme also works for more Verifiers), an Issuing/Opening Party and the Helping Party. The Helping Party resembles a Certificate Authority and provides the "verification table" (a form of public certificate) for the Verifier. Signer 1 and Signer 2 receive two different signing root secrets from the Issuing Party ($MemberSecret_1$ and $MemberSecret_2$, see Table 4.6. Signer 1 and 2 generate with their $MemberSecret_1$ and $MemberSecret_2$, respectively, new secrets for MAC generation. There are many methods to generate secrets from the member's initial secrets, but the recommend way is the generation of top-down hash-tress due to the low complexity for both, the Signers and the Helping Party, as described in the previous Sections. The secrets will be generated each time a time slice expires. Time slices may be freely defined and may overlap. The time slices are based on the common clock of the Signers and the Helping Party. There are several methods to synchronize the clock between parties, thus it should be noted that the method of synchronization does not affect the rest of the scheme. For example the method of [PCTS05] is adequate for constrained devices and could be used. The Helping Party provides the verification table, see Figure 4.31, which contains hashed values of the valid member secrets and the member secrets

themselves. While the hashed values are published when valid, i.e. according to their assigned time slice, the member secrets themselves are published after they expired. The reason for this is that the hashed values are used to verify a MAC for a message, while the clear text secrets themselves are used to verify that a Signer knew the initial secret and that he did not just steal the hash values from a published verification table.

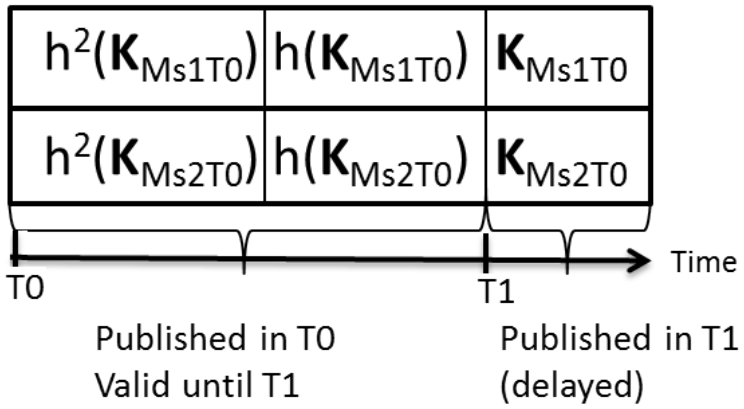

Figure 4.31: Example of a verification table

The overall relationships of the participants are as follows: every Signer generates secrets according to defined time slices. Whenever a time slice expires, a Signer will generate other secrets, for example by means of the canonical jump or other alternatives. The verification table provided by the Helping Party also depends on the common clock shared with Signers and the same time slices. This means, that if a time slice expires, entries in the verification table change as well. To request the verification table, the Verifier and the Helping Party have to authenticate each other. Every Verifier and the Helping Party share a unique secret KV_i. The verification table is encrypted per request by the Helping Party with the secret that is shared with the requesting Verifier. Figure 4.32 depicts the scheme for a Verifier and two Signers, Signer 1 in a time slice T0 and a Signer 2 in T1. The following secrets derive from Table 4.6 and Figure 4.30. The verification in this scheme work as follows at time T_i:

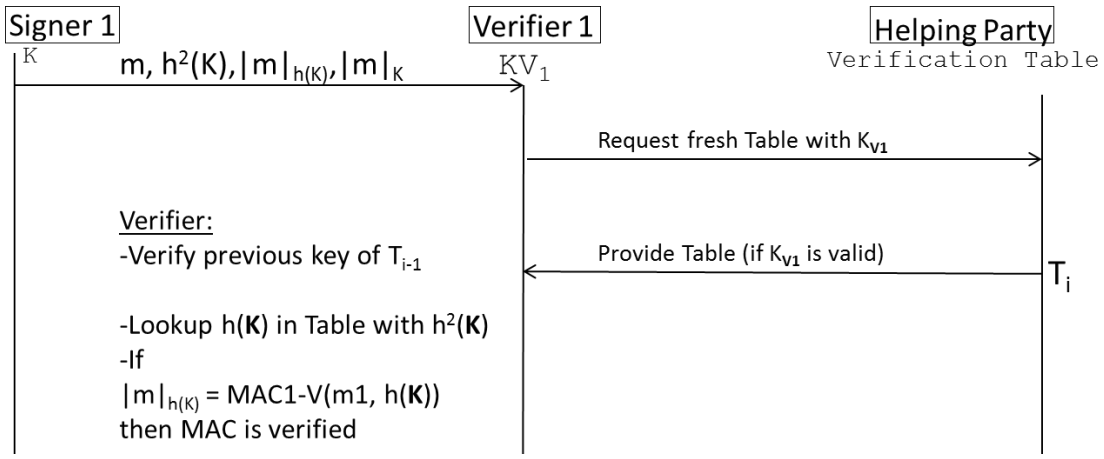

Figure 4.32: Simplified sequence diagram for the proposed scheme

Figure 4.32 exemplifies the group MAC protocol for a Signer 1, a Verifier 1 and the Helping Party. Although the scheme is simplified to show only one message and one Signer, the same mechanism would apply for several messages and several Signers. In the following the GroupMAC protocol referencing the secrets and the distribution from Figures 4.30 and 4.31 is described.

Algorithm 16 Actions of Signer 1

1: Given are m_1, $MemberSecret_1$, T_i.
2: **procedure** GETNEWSECRET($MemberSecret_1$, T_i)
3: $K_{Ms1Ti} \leftarrow$ *canonical jump* ($MemberSecret_1$,T_i)
4: Return K_{Ms1T0} ▷ Assuming T0.
5: **end procedure**
6: Prepare GroupMAC for a message m_1.
7: **procedure** GROUPMAC(m_1, K_{Ms1T0})
8: MAC1h $\leftarrow$ MAC(m_1,$h\,(K_{Ms1T0})$)
9: MAC1 $\leftarrow$ MAC(m_1,K_{Ms1T0})
10: $h^2\,(K_{Ms1T0}) \leftarrow h(h\,(K_{Ms1T0}))$ ▷ h^2 denotes double hashing.
11: Return $MAC1h$, $MAC1$, $h^2\,(K_{Ms1T0})$.
12: **end procedure**
13: Signer 1 $\rightarrow$ Verifier: m_1, $MAC1h$, $MAC1$, $h^2\,(K_{Ms1T0})$.

The steps for Signer 1 are the following. Signer 1 generates the first secret K_{Ms1Ti} from $MemberSecret_1$ according to the current time slice T_i. It is assumed that this is the first time slice, thus K_{Ms1T0} is generated (lines 3 and 4 in Algorithm 16). Signer 1 wants to send now a message m_1 to the Verifier with a corresponding group MAC. The Signer first generates a secret K_{Ms1T0} (the time slice T0 is assumed). The Signer then computes two message authentication codes denoted $MAC1h$ and MAC for the same message by using (i) the hash of his first

secret and the secret itself (lines 8 and 9 in Algorithm 16). Finally, the Signer generates a hash of the hash of his first key. Note that double hashing is denoted subsequently as h^2.

The Signer has now all the information that is needed to send a Group MAC protected message to the Verifier. Signer 1 sends the message m, and the Group MAC consisting of $h^2(K_{Ms1T0})$, $MAC1h$ and $MAC1$. The steps for Verifier 1 are as follows:

Algorithm 17 Actions of Verifier 1

1: Given are K_{V1}, m_1, $MAC1h$, $MAC1$, $h^2(K_{Ms1T0})$.
2: Verifier $\rightarrow$ Helping Party: K_{V1}; request for Verification Table.
3: Verifier $\leftarrow$ Verification Table. ▷ From the Helping Party, if K_{V1} is valid.
4: Look up $h(K_{Ms1T0})$ with $h^2(K_{Ms1T0})$ ▷ See Figure 4.31.
5:
6: In T0:
7: **procedure** VerifyGroupMAC(m_1, $h(K_{Ms1T0})$, $MAC1h$)
8: MAC1-V $\leftarrow$ MAC(m_1, $h(K_{Ms1T0})$)
9: **if** MAC1-V == MAC1h **then**
10: Return $Valid$. ▷ Group MAC is valid for m_1.
11: **else**
12: Return $Invalid$. ▷ Group MAC is not valid for m_1.
13: **end if**
14: **end procedure**
15:
16: In T1: ▷ K_{Ms1T0} will be available in T1 on the Verification Table.
17: Look up K_{Ms1T0} with either $h^2(K_{Ms1T0})$ or $h(K_{Ms1T0})$.
18: ▷ See Figure 4.31.
19: **procedure** VerifyKnowledgeOfSigner(m_1, K_{Ms1T0}, $MAC1$)
20: MAC1-V $\leftarrow$ MAC(m_1, K_{Ms1T0})
21: **if** MAC1-V == MAC1 **then**
22: Return $Valid$. ▷ The Signer knew K_{Ms1T0} in T0.
23: **else**
24: Return $Invalid$. ▷ The Signer did not know K_{Ms1T0} in T0.
25: **end if**
26: **end procedure**

The Verifier receives the message and the Group MAC from the Signer. The Verifier needs to verify that $h^2(K_{Ms1T0})$ is known to the Helping Party. The Verifier requests the verification table provided by the Helping Party with his secret K_{V1} and looks up h^2 with h(K_{Ms1T0}) (lines 2 to 4 in Algorithm 17).

The Helping Party computes the secrets in the same way as the Signers, e.g., by means of the canonical jump and according to different time slices, and creates

the verification table. The verification table (see Figure 4.31) is similar to the group public key. It is created with a round robin schedule in the steps of the Helping Party:

K_{Ms1T0} is a valid secret of a specific time slice. It is used to generate the hashed values $h^2(K_{Ms1T0})$ and h(K_{Ms1T0}) in the verification table (for details on the generation, refer to Algorithm 16).

When a time slice expires, new entries for valid secrets in time slice T_i are provided (e.g., $h^2(K_{Ms1Ti})$ and h(K_{Ms1Ti})).

When a time slice expires, the original input for hashes, in the case above K_{Ms1T0}, is revealed in the table.

The validation of the MAC and the authenticity of the message is given, if the Verifier can compute MAC(m, h(K_{Ms1T1})) himself. Furthermore, the Verifier can verify that the Signer knew the original secret K_{Ms1T1} after it was revealed, by computing MAC(m, K_{Ms1T1}).

Figure 4.32 displays the protocol for one Signer, but a second Signer would do the same with his chain of keys, e.g. $K_{Ms[j]T[i]}$ for a Signer j in time slice i. Additionally, Figure 4.33 shows that the Verifier is able to retrieve K_{Ms1T0} (the original non-hashed secret) from the verification table in T_1, and to verify that the first Signer in T_0 knew the original secret.

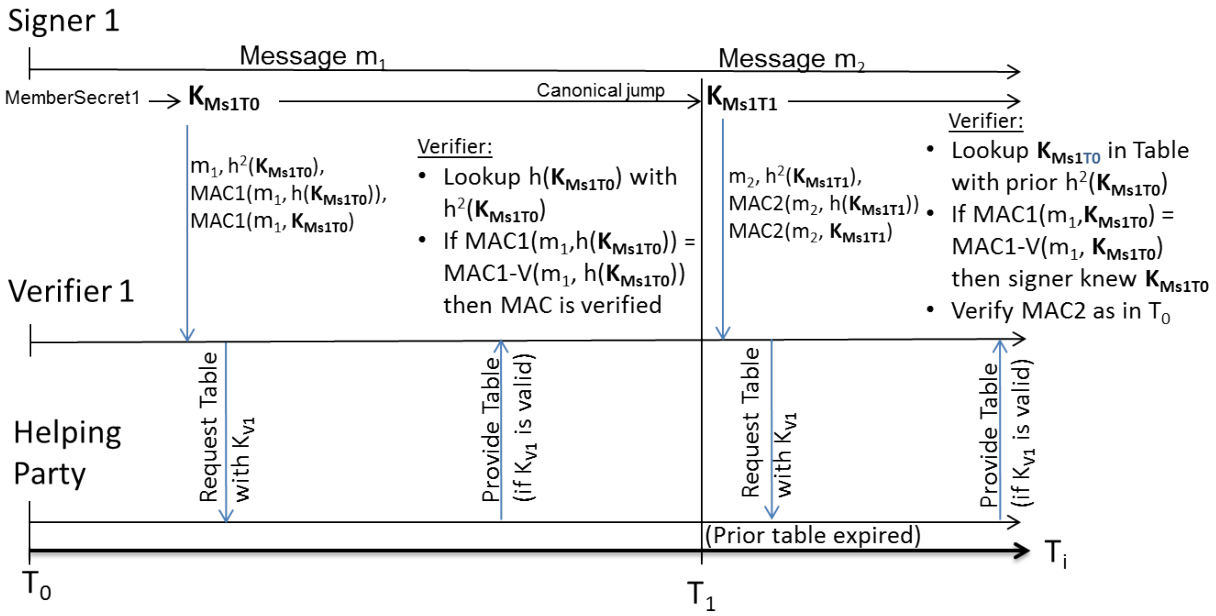

Figure 4.33: Verification of two messages in two different time slices

4.3.4.5 Considerations and Optimization of Group MACs

In this subsection considerations on the design decisions as well as the privacy and implementation are discussed. For the formal analysis of the protocol please refer to the annex, Section B.4. The Sections ends with a comparison of the original group signatures scheme and Group MACs.

Considerations on the public verification table. The scheme provides a static table that is provided for the length of a time slice. A Verifier is able to download and store the verification table, thus being able to stay disconnected form the Helping Party. Additionally, the verification table only displays a few hashed values per Signer, thus staying manageable for the Helping Party and the Verifier. Several optimizations can be made to the verification table as well. The Verifier does not have to download the whole table:

If the Verifier looks up hashed values dynamically, the Helping Party could provide only one or a few values instead of the whole table.

Verification tables can be divided into smaller parts, depending on the Signers' properties (e.g. Signers assigned to the property "legal age" are in one group, etc.).

Important note: a Verifier needs to be semi-trusted to obtain a Verifier Key K_{Vi}. In case a malicious Verifier exploits the hashed values of the verification table, he is able to sign messages for the time slice assigned to those values. Once the time slice expires, Verifiers will notice that the malicious Verifier could not provide a MAC with the original non-hashed secrets and trigger the replacement of all Verifier keys, excluding those Verifiers that are not fully trusted.

One possible solution to identify a malicious Verifier is to provide a verification table with marked secrets. Honest and fully trusted Verifiers might then identify the marked values and notify the Issuing Party to trigger a revocation.

Revocation of secrets. As in the original scheme, the Issuing Party is proposed as the revocation authority. Revocation of secrets is generally done by updating secrets, which are used in the MAC generation and verification processes, and by omitting certain values in the verification table. In case of revocation of a

Signer, the Helping Party will stop publishing the Signer's hashed values in the verification table. In case of revocation of a Verifier, the Issuing Party will notify the Helping Party, excluding the Verifier from further verification (particularly, denying any new table requests). In case of revocation of the Helping Party, all member secrets and Verifier secrets have to be refreshed. In case of revocation of the Issuing Party, a new Issuing Party has to issue new keys to all other parties. Additionally, it has to recreate the member-secret issuing table.

Comparison between Group MACs and Group Signatures. Group MACs resemble the properties of group signatures (as identified in Section 4.3.4.2) in the scheme the following way:

Separation of duty. In Group MACs, several parties fulfil different obligations as well. Neither the Verifier, nor the Issuing Party nor the Helping Party can deanonymize MACs by themselves. For linking a MAC to a Signer, the Issuing Party lacks pairs of messages and MACs, which it does not have access to. The Verifier lacks the member-secret issuing table and the member root secrets. The Helping Party lacks messages, MACs and the member-secret issuing table.

Unlinkability. Neither party is able to link a MAC-Signer pair by itself, even after prior re-linking of one or several MACs to a user. This is due to the nature of the top-down hash-tree and the canonical jump, the Verifier is not able to predict or recognize a given secret, even if it knows the preceding one. The Helping Party, as well as the Issuing Party, lack knowledge of message-MAC pairs and are not able to link messages at will.

Revocation of secrets. The Group MAC scheme allows revocation of all participants' secrets, as elaborated in the respective paragraph above.

4.3.4.6 Protocol Verification

Details regarding the verification of the GroupMACs protocol and the used formal and technical tools are given in Sections B.2 and B.4.

4.3.4.7 Summary

The Group Message Authentication Code scheme offers a novel approach to authentication with privacy, that is purely based on symmetric cryptography and hash-functions. This allows using the scheme in situations, where resources are restricted or where no asymmetric computation is possible. The approach relies on the top-down generation of hash-trees explained in Section 4.3.3.4, which allows the efficient generation of virtually unlimited secrets. Hash-functions are already available in even the most constrained devices, thus allowing the implementation of the scheme under the constraints of Section 2.6.2. In comparison, the Group Message Authentication Codes scheme resembles the properties of Group Signatures under the assumption of a semi-trusted Verifier and an active Certificate Authority (or Helping Party) providing a dynamic, changing and public verification table. This assumptions exclude the Group MACs scheme form any scenarios that offer public verification of signatures or MACs, but is seamlessly applicable to the impact areas of Section 2.1.

Chapter 5

A Privacy Enhanced IoT Domain Model

In this Chapter, the Rerum domain model presented in Section 2.4.3.1 is extended by a total of four new components: (i) Consent Management as presented in Section 4.2.2, (ii) Privacy Policies as presented in Section 4.2.3, (iii) Access Control Policies[1] and (iv) Trust, see Section 5.1.1. This resulting domain model is Rerum's final domain model. Note that privacy enhancing technologies like the privacy dashboard, geo-location privacy PET, MallMACs, GroupMACs are a part of the instantiation of the domain model, as explained in the respective sections and, on a conceptual level, in Section 2.5.4.

The Rerum final domain model is then extended again with all concepts and technologies that were not integrated in Rerum, in particular (i) Pseudonym Management and (ii) Ownership. The *privacy enhanced IoT domain model* in Section 5.3 implements all contributions presented in this thesis.

5.1 Rerum Pre-Final Domain Model

This section shows the pre-final Rerum domain model. Note that the model shows an incomplete mapping of all concepts and technologies that have been presented in this thesis, as the Rerum project ended while certain technologies were still in the verification stage.

[1] Access control is part of the security functional components, the interested reader is referred to [RER15].

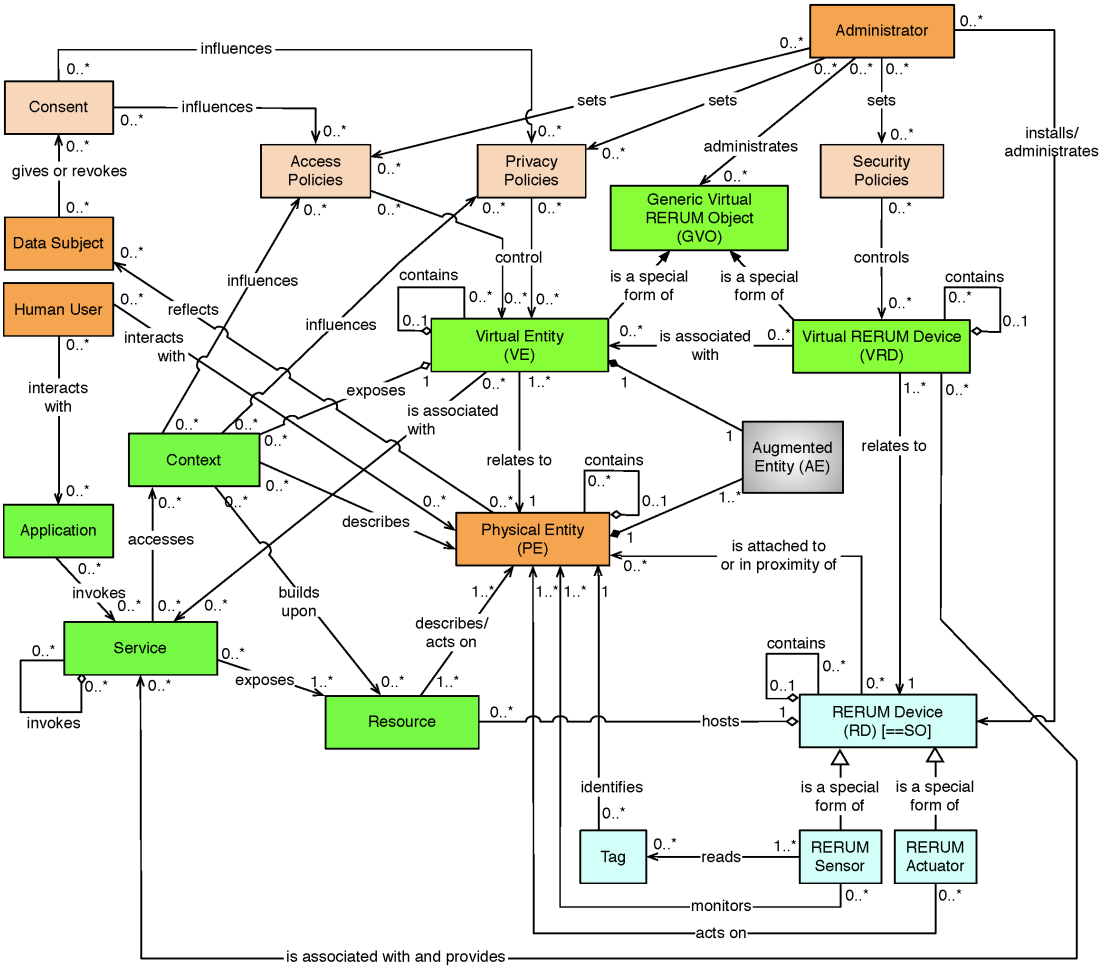

Figure 5.1: Rerum domain model - six new extensions

The pre-final model presented in Figure 5.1 updates Rerum's first phase model discussed in Figure 2.7 by seven components. The reader is referred to Section 2.4.3.1 for a full description of the main elements and concepts of the first phase.

Data Subject. The term data subject was introduced in earlier section with the note that throughout this thesis, a "user" is always defined as a data subject. Rerum differentiates the types of human users that participate in IoT, one of them being the data subject. The data subject is a special user that has a special relation with the entities around him (the entities represent or disclose some information about him). The data subject can give or revoke consent and therefore affect access and privacy policies.

Consent. Whenever an entity represents or discloses some information about a human user (in particular: a data subject) and a service provider (in Rerum

terms: an app) is interested in processing that data, the concept of consent is involved. Consent affects the question of "who" and "how" regarding the access to information of Virtual Entities. As access information can be represented as policies, consent has direct influence on access control and privacy policies. This component and its relationships document the discussion in Section 4.2.2.

Security Policies, Access Policies and Privacy Policies. Rerum defines three types of policies: (i) security policies that control access to virtual devices, (ii) access policies and (iii) privacy policies that control the access to Virtual Entities. Privacy related policies are defined by the data subject through consent management or directly with the privacy dashboard. Additionally, context maybe an integral part to access decision, e.g. when an user is at work, at home or in an emergency situation. This concepts document the outcome of Sections 4.2.3 and 4.2.2.

The model does not capture specific technologies like sticky policies and the PAT protocol, which are a part of the technical view in the Rerum architecture. Rerum defines one final extension to the domain model for trust. Before the final Rerum domain model is presented, the following section takes a brief look at how trust affects security and privacy and why it has to be implemented for a privacy enhanced domain model in IoT.

5.1.1 Trust and Privacy in IoT

Rerum additionally defines trust relationships for the final domain model. In general, trust in IoT can be exemplified with access control: compliance with access control policies cannot be forced and relies often on the expected behaviour of the parties involved. Trust and reputation are highly related to security, but they can also contribute to other factors such as reliability, availability and privacy, see [YZV14]. In other words, access control needs to be based on trust to be adequate for privacy, for IoT in particular. Gambetta defines trust as the subjective probability by which an individual, A, expects that another individual, B, performs a given action on which its welfare depends, see [G+00]. In web systems, trust is often a binary probability that is linked to authentication. In a

typical access control scenario, a user is fully trusted if he is able to authenticate against a service. The service is then able to decide on the user's requests, because the service knows the role he is assigned to. Newer access control methods, such as single-sign-on, do a variation of authentication and role assignment, but the underlying method is the same: a user is trusted if he is able to authenticate and his role is known to the system. In IoT, users are not predetermined and thus cannot be known a priori by the services. Services themselves might be created dynamically including their own access control components. Additionally, services might be constrained and thus not able to store a high amount of user IDs and user roles. The assignment of rights to users has to be done dynamically, that means, if an unknown user wants to access a system, a mechanism has to reason about if the user is allowed to access a service or not. This can be done through delegation and trust management, as proposed in [KFJ01]. A user that is trusted by the system can delegate all or a subset of his rights to another user that he trusts and that can fulfil the requirements associated to the rights. Delegated rights can be access and delegation rights. With delegation rights, the users can generate a trust chain. Every member of the chain can access the system according to his respective rights until they expire or until the chain is broken. This is the case when a user becomes untrusted or fails to meet the requirements associated with a delegated right. In that case every user in the chain will not be able to exercise his rights, and the chain has to be created anew. How users and the service quantify trust may be different. Depending on the context, they might share secrets that define their trustworthiness or they might evaluate trust based on reputation. Reputation is the collected and processed information about a user's former behaviour as experienced by others. This is especially interesting due to the heterogeneity of users in IoT.

5.2 The Final RERUM Domain Model

The Rerum final domain model shown in Figure 5.2 complements the model in Figure 5.1 with the concept of trust as presented in the Rerum final system architecture deliverable [RER15]. Trust and reputation, as elaborated above, affect several concepts of the IoT domain.

Trust influences access control decisions and thus the way access and privacy policies work. Past (mis-)behaviour is documented in reputation schemes, which characterise what to expect form devices. The same mechanisms apply to users, be it applications and services or human users. Misbehaviour covers intentional and unintentional errors in measurements and threatening misbehaviour from human and digital users.

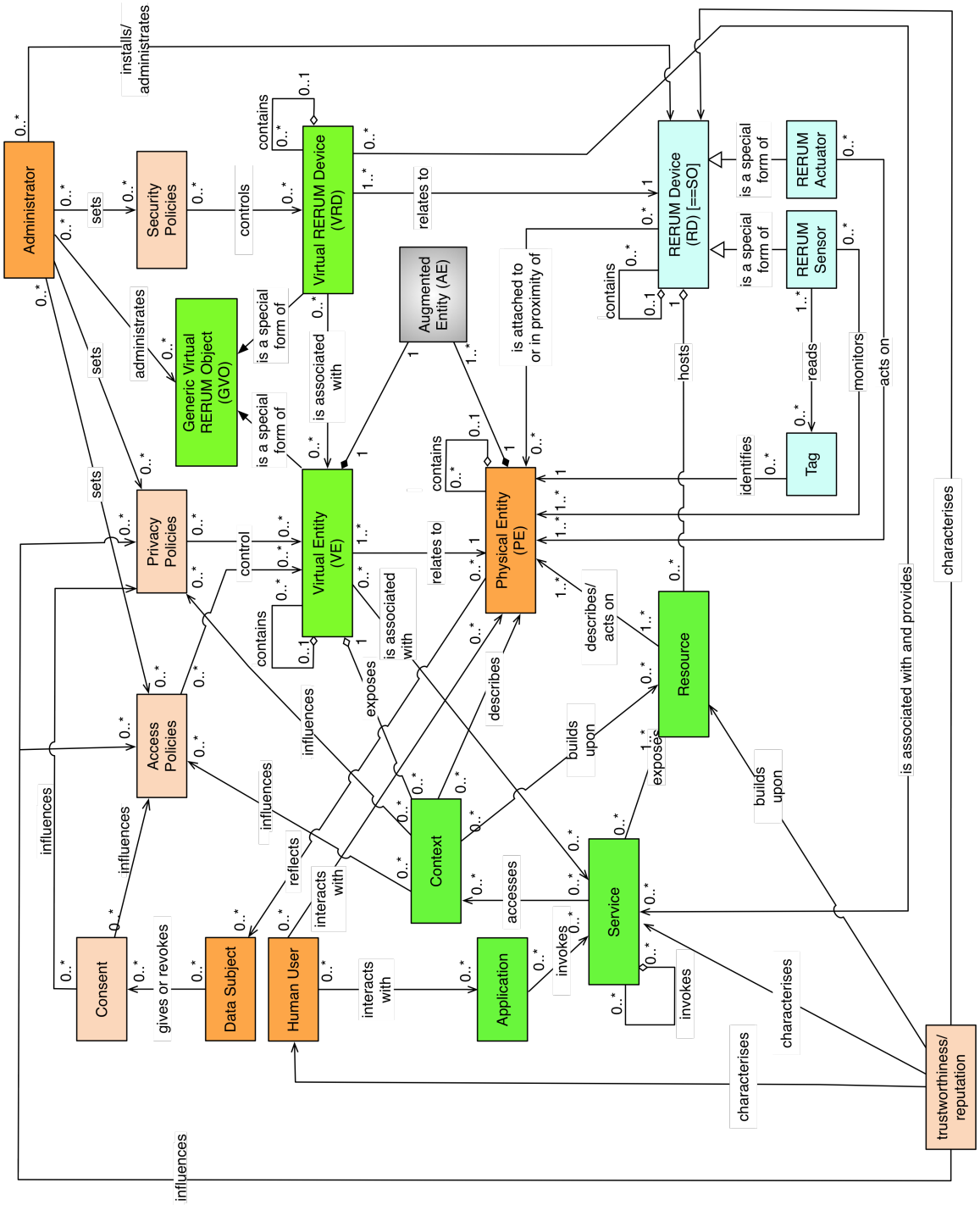

Figure 5.2: Rerum final domain model

5.3 A Privacy Enhanced IoT Domain Model

The privacy enhanced IoT domain model presented in this thesis is a key contribution of this thesis. The model updates Rerum's final domain model by two concepts: (i) ownership and (ii) pseudonyms. Ownership relationships were explained in Section 2.5.3. A new relationship emerges for the Data Subject. A Data Subject is reflected by zero to many Physical Entities. This means that zero to many Physical Entities relate to a human user, which may relate either to absolute ownership or physical possession. An exception would be a human user that is sensed by an entity in his physical proximity. This would only satisfy the reflection relationship for the data subject without ownership.

Pseudonyms are the main form of identification in a privacy enhanced IoT domain model. Pseudonyms are generated by all Virtual Entities, thus pseudonyms are associated to the abstracted class of Generic Virtual Rerum Objects. Devices are able to communicate by themselves and are able to generate pseudonyms as well. Pseudonyms are an integral part of communication, as they serve as the host and identity identifiers in IoT. Thus, the relationship is modelled as one-to-many for the aggregation. One to many because if an entity or device exists, it has at least one pseudonym by with it is identified.

As elaborated in Section 2.5.3, ownership affects the assignment of identities. The reason is, that an owner must be able to recognize his devices while other services should only temporarily be able to track a set of pseudonyms. Hence, a relates-to relationship is defined. Figure 5.3 presents the complete model.

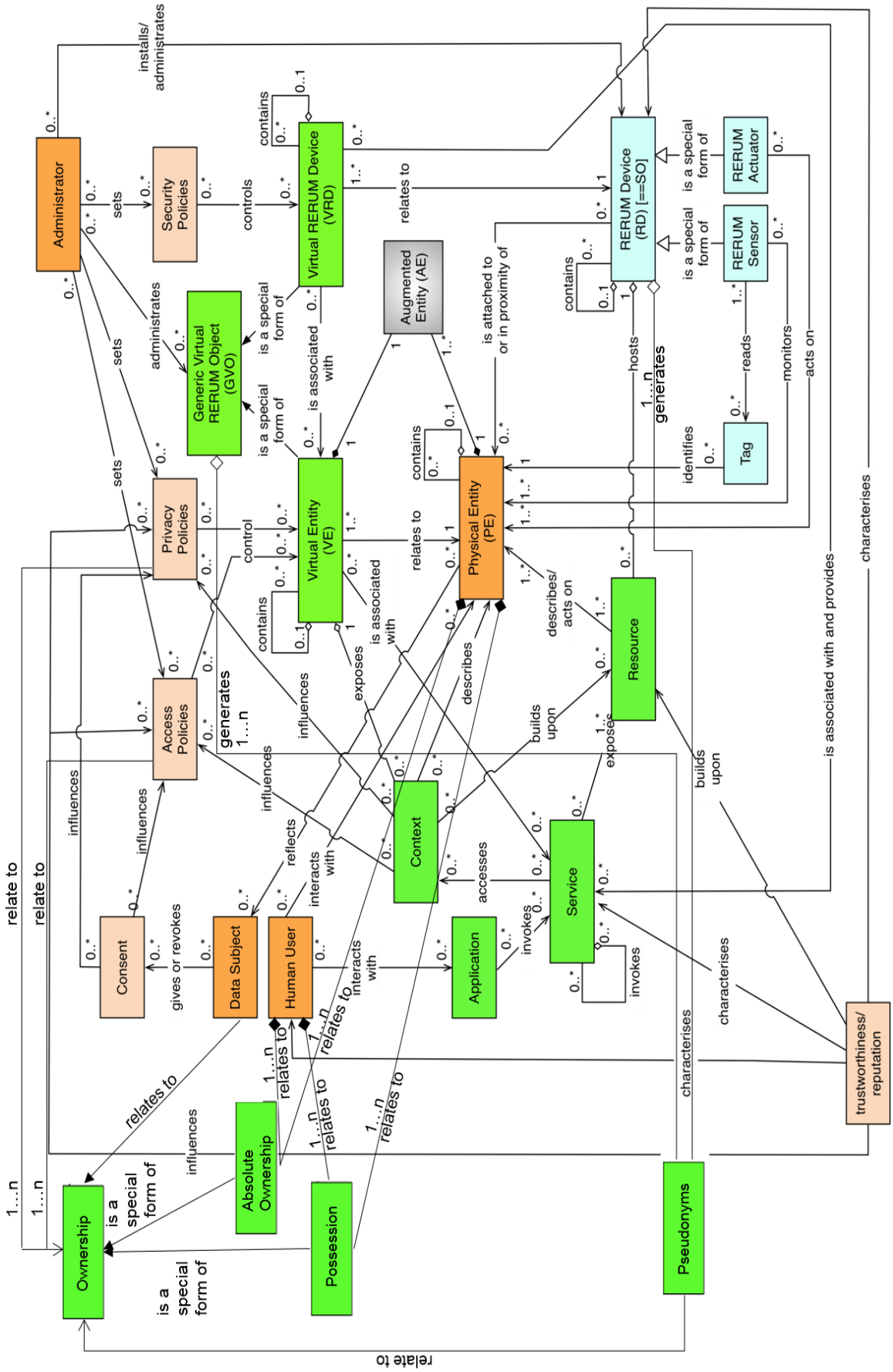

Figure 5.3: A Privacy Enhanced Domain Model for IoT

5.4 Summary

The Rerum domain model encapsulates security, privacy and trust (SPT) concepts developed for Rerum. The enhanced privacy domain model furthermore extends Rerum's domain model with two additional privacy concepts. Notably in the final result, privacy components are well represented with four out of seven SPT components. The relationships affect all fundamental IoT components, either directly or indirectly (e.g., devices are affected by their virtual counterpart). This shows how influential privacy is for every aspect of IoT and how it needs to be postulated in order to provide adequate data protection. For future practicability, following assumptions hold:

In general, both model and architecture are living artefacts and will change over time. They may be extended, updated and adapted along with IoT's growing popularity and changing technologies.

The architecture is an instantiation of the domain model. While the domain model is a fundamental set of building blocks and is less prone to change, the architecture is strongly affected by current and future technologies, intended operational environment, regulations and even personal preference (of architects and developers).

From the perspective of an IoT system developer, the domain model is a fundamental part that is taken "as is", while the architecture is freely adaptable. The Rerum architecture was implemented as an open source project[2] and can serve as a starting point, but developer guidelines on design decisions, recommendations and quality management procedures maybe further needed. This is considered as a possible next step.

[2]See https://github.com/ict-rerum, last visited August 17, 2017.

Chapter 6

Conclusion

The motivation of this thesis was presented in Section 1.1. This thesis addresses two seemingly opposing topics, the Internet of Things which centrates on collecting and processing data, and privacy, which prominently propagates data minimization. In order to grasp if these topics are compatible and to foresee a positive result ("yes they are, under this conditions") or a negative one ("no, a conflict between service providers, legislation and users is to be expected") three main objectives were documented.

The first objective was the identification of building blocks of the Internet of Things. At the time of this writing, there was neither a consensus on what IoT really is, nor what core elements (building blocks) it is based upon. Also, although IoT is very popular, many of its aspects (economical, technical and social) have not been made clearly transparent. As a first step, a literature review was carried out to understand the motivation behind IoT, its economical value and the most promising building blocks for future IoT systems. The result was an outline of available technical standards and architecture proposals, a reconciliation of the building blocks of IoT systems and a holistic definition of IoT. While the method of identification of building blocks was taken from the IoT-A and Rerum projects, this thesis was able to verify the methodology by applying its holistic definition to the project's results.

With adequate building blocks, several aspects could be analysed. First, the economical motivation in different IoT impact areas were elaborated. This step was important in order to identify assumptions and intended environments of

privacy technologies. Also, “tradtional” ICT systems and IoT systems have been set into relation to get a clear understanding of similarities and differences. Form a privacy perspective, the participation of system users can be clear-cut: in traditional ICT systems, users are active participants of a system, their personal data is introduced by themselves (knowingly or unknowingly). In IoT, non-users maybe included in systems and their personal information maybe introduced unknowingly. The cyber-physical property of IoT was also recapitulated. The relation between physical and virtual representation opens several questions that have only received little attention in computer science (“social IoT”), but have been extensively discussed in social sciences. Therefore, this thesis introduced the concept of ownership in IoT, a concept that has been a building block in jurisdiction and modern western societies. The main contribution targeting the first objective was a reference model for IoT, accompanied by a set of assumptions, requirements and constraints, both technical and economical.

The second objective was to survey relevant privacy requirements, concepts and technologies. The relevancy was measured upon IoT impact areas, IoT building blocks, technical and economical assumptions and constraints. In the process, the fundamental conflict of privacy and IoT became present and was recapitulated. Several privacy principles, standards, taxonomies and frameworks were discussed. Requirements stem in particular from legislation GDPR and good privacy practices (privacy by design). After providing an overview of how those apply to a general privacy engineering methodology and adopting a privacy taxonomy for the rest of this thesis, a privacy lifecycle development was proposed to find a reasonable approach and set of existing proposals. The lifecycle was developed such that it supports the fulfilment of identified privacy requirements. Working with Rerum, an overall lack of privacy building blocks and technologies that support the lifecycle was singled out, therefore some privacy building blocks and their integration in an IoT architecture was presented. In particular, privacy policies, consent management, pseudonym management and ownership were added to the IoT domain model.

As for the technologies, economical and technical constraints impede traditional PETs from being applied to IoT. Additionally, PETs tend to use special types

of arithmetics that are not available off-the-shelf and would cost additional implementation, testing and maintenance effort. Thus, in an extended effort, technologies for efficient PETs were surveyed. In this thesis it was demonstrated and formally verified that, although none of the existing technologies satisfy the requirements set in IoT impact areas, off-the-shelf cryptography can help to build further PETs for constrained devices. Furthermore, it was shown that asymmetric properties can be reached with symmetric cryptography under minor assumptions. The results presented hereby are merely the tip of the iceberg and seem to hold significant potential. The PETs were elicited based on scenario descriptions of the Rerum project and aim to support the proposed PDLC. The technologies were included conceptually in the instantiation of the domain model, namely in the Rerum architecture.

The last objective sought to combine the IoT and privacy building blocks in a privacy enhanced domain model. This domain model is a fundamental reference that describes relationships and dependencies of a privacy enhanced IoT system. The domain model, accompanied by the technologies, formulated assumptions and constraints of previous steps, is one of the main contributions of this thesis.

The three objectives passed demonstration, evaluation and communication phases respectively. Demonstration and evaluation of scientific and industry aspects were achieved through publications and patents in collaboration with the Rerum project and Siemens AG.

6.1 Outlook

Although many new insights about privacy in IoT could be identified, many questions remain open.

For instance, the identified impact areas in this thesis are just a subset of all possible applications of IoT, some may become evident in the future. Although diverse, data generation in these areas follow many automatisms where devices are programmed or follow policies to automatically collect data. Consequently, they sense the status of anything that occasionally happens in the devices' environments, observing users' activities either directly or indirectly. The question in focus is

how these new impact areas will change the privacy enhanced domain model. Here, computer science and jurisdiction have shown to have strong synergies, as was the case of ownership.

The enhanced privacy domain model itself has not been implemented and is considered as future work. Although the Rerum domain model was firstly implemented in the Rerum architecture, some significant concepts and technologies could be verified after the project's end.

Also, developer guidelines on design decisions, recommendations and quality management procedures are further needed. Additionally, the application layer of the IoT stack has been widely left out of scope and needs to be addressed.

Regarding technologies, the assumptions and the constraints that they were designed to work in, this thesis showed that in particular economical constraints hardly change, and if so, on a very slow pace. Thus investments in integration of privacy technologies (as part of industrial R&D) are limited, as R&D costs are tightly controlled, even for highly digital products. On the other hand, technology constraints loosen up. Nevertheless, fancy arithmetics as required by many academics will not be implemented in the near future, due to possible high implementation, quality management and maintenance costs.

Ownership and consent have been shown to be prime examples of interdisciplinary research. For future researchers interested in IoT, the readers are advised to consider interdisciplinary work as IoT and computer science have enlarged their scope well into other research areas.

The introduced technologies for efficient PETs that have been researched and pointed out can help to build further PETs for constrained devices.

Privacy in IoT remains difficult, because privacy engineering itself is a task that still needs to be well understood. Privacy engineering lacks a consensus of best practices and supporting tools. The need for more efficient PETs suited for constrained environments and in particular tools that support transparency for users, allowing them to control and access their data throughout multiple providers and processes, seems evident.

Concluding, this thesis and the acceptance of its results by several parties shows that IoT and privacy are not disjunct topics. There is justifiable interest in

IoT, as it helps to tackle many future challenges found in impact areas. At the same time, IoT impacts the stakeholders that participate in those areas, creating the need for unification of IoT and privacy.

As was shown in the course of this work, technical and economical constraints do not impede the unification of IoT and privacy, although this process has merely begun. Thus further work towards this goal is still needed.

Appendices

Appendix A

Assumptions on the Virtual Representation of Users

This interview was conducted on the 11.04.2014 with Dr. Jorge Cuéllar Siemens AG. At the time of the interview, Dr. Cuéllar was aware of the ownership, privacy policy, consent and pseudonym concepts presented in this thesis.

Santiago R. Suppan: *Hi Jorge, thank you very much for your time. In this interview I would like to know your take on the virtual representation of a user in IoT. The main idea is to have a digital space where the user can manage his identity, his policies, his consent, pseudonyms etc. My first idea would be to take his smartphone. Smartphones are an ubiquitous, personal devices that many users have. But for the smartphone to uniqueliy identify the user, the user would have to authenticate himself by some means that shows that he is either the owner or possessor of the smartphone, so it can load the correct attributes, policies, etc.*

Dr. Jorge R. Cuéllar: *I agree on the smartphone as the principal device for representing the user according to the [IoT-A] model. But I don't think the smartphone needs to recognize ownership and possession. All of us have a smartphone and only very seldom do we lend or share it. We can assume ownership for a user and his smartphone.*

Santiago R. Suppan: *What about if a user accidentally loses his smartphone? He will be excluded form an IoT system. If the user could uniquely identify himself with an RFID tag plus another factor, he could use a different smartphone quickly.*

Dr. Jorge R. Cuéllar: *And how seldom does that happen? Of course two factor authentication is a good thing, but it is not necessary for ownership here. I, and many other colleagues form the telecommunication area, think that*

the smartphone is a very personal device. Users take well care of it. If the participation of a system is essential, the user could have two or more smartphones as backup.

Santiago R. Suppan: *I see. The smartphone fully represents the user, such that other devices recognize ownership and possession from their relationship to the smartphone, but not to the human user himself.*

Dr. Jorge R. Cuéllar: *You could see it like that. Something may change depending on future technologies, but there is a strong assumption that [the smartphone] will play a major role for IoT in the near future.*

Santiago R. Suppan: *Thank very much.*

Dr. Jorge R. Cuéllar: *You are welcome.*

Appendix B

Protocol Design and Verification

B.1 Design of Efficient Protocols

Note: the protocols defined in this thesis follow good design principles from the IETF and Ryan et al., which are not further discussed here. The reader is referred to RFC 1958 [Car96] for general good design principles of internet protocols and [RSM+01] for modelling of security protocols. The subject of this section is exclusively efficiency.

Many privacy enhancing technologies are based on asymmetric cryptography. Asymmetric cryptography is a general term for crypto systems that are based on integer factorization, the discrete logarithm problem and elliptic curve relationships. The cryptographic primitives based on these problems have been widely studied and have become highly efficient over time. The problem with asymmetric primitives for IoT is two-fold:

Asymmetric primitives often need different arithmetics. For example, the most efficient key exchange scheme is ECDH, see [BJS07]. The most efficient and practical privacy friendly authentication scheme is the Short Group Signature scheme, see [BBS04]. ECDH is based on elliptic curves and the Group Signatures are based on bilinear groups. A constrained device would either need two crypto-libraries or two crypto-processors to use both. Group Signature schemes based on elliptic curves exist, but they lack practicality due to drawbacks in revocation and key length, as seen in [HWL04]. There might be also differences in the properties of elliptic curves, therefore it is still difficult to have one common arithmetic.

"Off-the-shelve" libraries and crypto-processors supporting privacy enhancing technologies based on asymmetric primitives are rare. While crypto-processors exist with, e.g., elliptic curve or bilinear group arithmetic, they do not support calling atomic functions. Rather, they provide a transparent interface for signing or encrypting inputs, without revealing the low-level (atomic) processes. This

means that a crypto-processor for, e.g., group signatures has to be manufactured, which is a high-cost endeavour. Software libraries exist, such as the PBC library[1], but significant effort would be needed to optimize the library for constrained smart objects with class 1 limitations.

The reader is referred to RFC 5218 [TA08] for more indicators on what makes a protocol successful.

A possible solution are protocols based purely on symmetric cryptography, that resemble asymmetric protocols. This has been done before, for example, Ralph C. Merkle proposed a public key infrastructure based on hash-trees, see [Mer79]. The original approach by Merkle is rather inefficient due to key management, but essentially faster for the authenticating parties. The scheme has found several revisions and its benefits have been recently been underlined, e.g., in smart city applications, see [LLZ+14].

Another example is the TESLA protocol [PCTS05]. The TESLA protocols resembles a broadcast authentication and integrity protection scheme, were the broadcaster is the only party to compute authentic and integer messages. This would be easy in an asymmetric context, but inefficient in low power and low cost devices. The protocol again uses hash-functions only to achieve its properties.

From both, the Merkle authentcation and the TESLA protocol, several benefits for IoT become evident:

Symmetric protocols are based on one-way functions. One-way functions, such as cryptographic hash-functions, are easy to implement, can be found in "off-the-shelf" crypto-processors and are up to 1.000 faster in computation and 100.000 times (compared with RSA, ca. 2.000 times compared to elliptic curves) more efficient regarding battery consumption, see [PRRJ03].

Due to the lower battery consumption and the "off-the-shelf" availability of symmetric protocols, the application of security and privacy protocols becomes possible in scenarios with strong limitations. The challenge of defining such protocols to resemble their asymmetric counterparts remains still an open challenge, as not all asymmetric properties can be mapped to symmetric methods, see for example [ANN06], or, if possible, are very inefficient, see [Dif88].

B.2 On the Verification of Security Protocols

The verification of security protocols aims to provide a statement about how and if an attacker can learn something during protocol execution. One possibility would

[1]The PBC library is a pairing based crypto library, provided by the Stanford University, see https://crypto.stanford.edu/pbc/.

be to implement a protocol and actively test the protocol to prove any flaw. As simple protocols can become very complex, a mathematical model that abstracts some details of a protocol can also be used. A model typically defines several assumptions about the used technology and the attacker model (time, economical resources, technical resources), such as the model of Dolev and Yao, see [DY83]. The so called Dolev Yao makes three abstractions: first, cryptography is assumed to be perfect, there is no way to attack the cryptographic schemes used in the protocol. Given an encrypted message, the attacker has to know the encryption key in order to decrypt the message and to learn the content of that message. This assumption salves the model from any technical and implementation details (and errors). Second, messages are always disclosed completely. There is no partial disclosure, as the attacker is only able to decrypt a whole message and not parts of it. Finally, Dolev and Yao give the attacker full control over the network. The attacker is therefore able to passively eavesdrop the messages of a given protocol, but also drop messages, change messages, repeat messages and create messages.

The Dolev Yao attacker model is considered to be the strongest attacker model to measure a protocol against [Cer]. With the attacker model and given a security protocol, several security claims can be derived. These claims should be tested throughout various protocol instantiations to prove that the protocol does not have obvious or transitive flaws. These is a complex task, therefore a branch of verification tools have emerged.

Mitchell et al. have shown in [MSDL99] that the statement about the security of a protocol, i.e., if an attacker is able to compromise a message in a given security protocol, is undecidable. Therefore an additional assumption has to be made regarding the number of execution that the protocol is bound to have. This simplifies the verification such that a statement becomes decidable, see [MS01]. Therefore, tools based on decidable models are appropriate tools for the verfication of security protocols.

For the verification of protocols in this thesis, work of Cremers et al. is used. A description of the used model, semantics, assumptions regarding decidability can be found in [CM05]. The model was chosen as its fundamental semantics cover the Dolev Yao attacker model and provide a high completeness when modelling pure symmetric protocols, see [CLN09] for a comparison. Additionally, Cremers et al. offer an implementation of their verification model with the Scyther Tool. A description of the tool, it's abstractions and assumptions regarding error tolerance of the implementation can be found in [Cre08].

Furthermore, a full description of the Scyther modelling language, the interpretation of outputs, available claims and tool parameters can be found in the

work of Chen [Che15] and the Scyther manual [Cre14b]. The following subsection explains the the scyther formal model in a nutshell.

B.2.1 The Scyther Formal Model

The scyther formal model can be described in five steps:

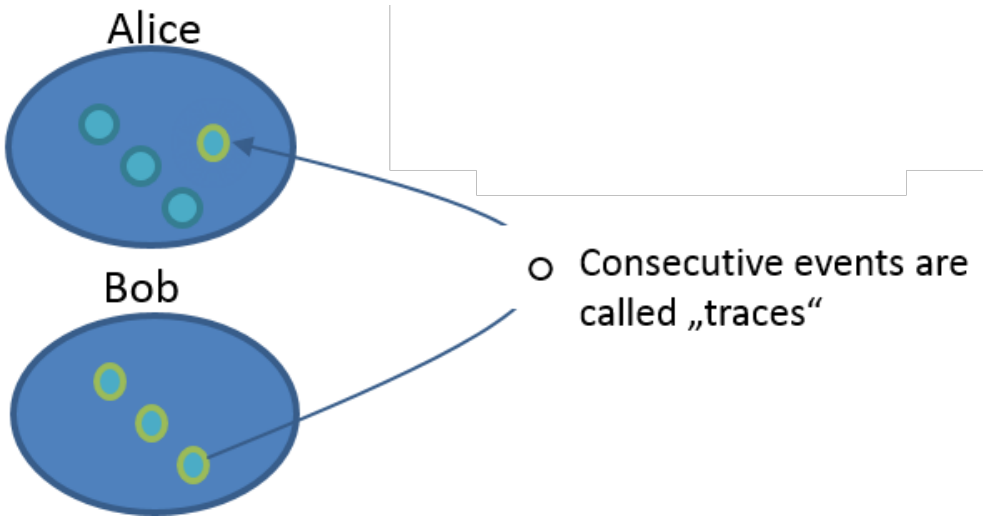

Figure B.1: Scyther - Basic formal model.

1. Roles are sets with elements. Elements can be variables, time information, random numbers, etc.
2. A protocol describes allowed operations (send, receive, compare, generate, cryptographic operations, etc.) in a specific order (so called traces) to change, unify add or disjunct sets. The result of a specific order is called "event".
3. The result of a protocol run are new Roles (sets with elements) which, according to claims, can only be reached with traces of the protocol.
4. Scyther tries to generate a trace that contradicts every claim. In particular it generates an attacker set.
5. Scyther considers multiple protocol runs to generate traces for the attacker (different instances by multiple agents).

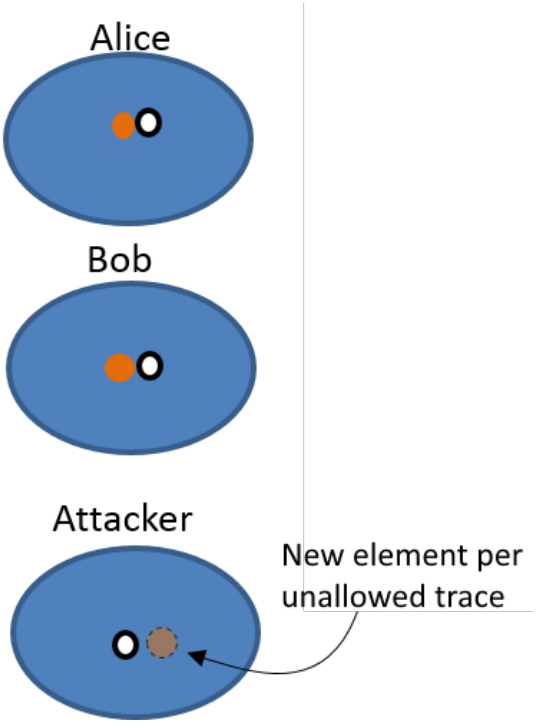

Figure B.2: Scyther - Protocol execution

Figure B.3 shows a possible attack trace over two protocol runs. Note: in the introduction of this section it was mentioned that the verification of protocol is only decidable, when protocol runs are limited.

According to the Scyther documentation, it is justifiable to assume that no further attacks a re possible if no attack where found within a bound of 5 protocol runs.

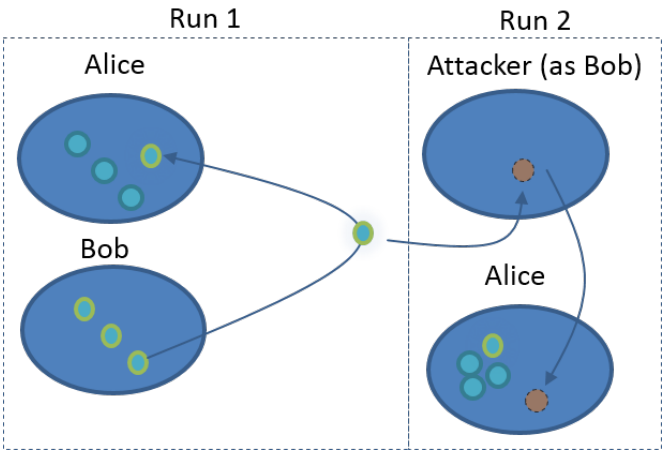

Figure B.3: Scyther - Matched typing and bounds

6. Scyther considered attacks that are based on type-flaws. A type flaw assumes that a Role was not able to recognize the expected message and may mix up messages of different types. For example, a nonce may be exchanged for a string by an attacker. A type flaw would take place if the receiving Role does not recognize the message swap. Note: in this thesis, typed matching was chosen, that means the assumptionthat the protocol implementation will impede type flaws.

B.3 Verification of the MallMACs Protocol

For the verification of the MallMACs protocol, Scyther linux version 1.13 was used on an Ubuntu 64Bit 17.04 machine with Python version 2.7.13. Note that the Scyther folder has to be assigned execution rights for all files. The Scyther configuration was *typed matching*, *5 protocol runs as maximum bound*, *10 maximum number of patterns per claim* and *find all attacks*.

The Scyther model file of the protocol MallMACs is as follows:

```
hashfunction hash;
usertype String;

// The path from X0 to X000 - Signer and Verifier know it,
// but makes no difference if it is public

protocol MallMAC(Signer,Sanitizer, Verifier)
{

role Signer
{
//m=f+a, f=messagefix+pathtor, ms is MAC of m with key S
fresh messagefix : String;
fresh a : String;
fresh pathtor: Nonce;
fresh pathtosz: Nonce;

// assure both parties are alive
claim(Signer, Running, Verifier, pathtosz,
hash(pathtosz, k(Signer, Verifier)));
send_0(Signer, Verifier, {pathtosz,
hash(pathtosz, k(Signer, Verifier))}k(Signer, Verifier));

var resp;
recv_01(Verifier, Signer, {resp}k(Verifier, Signer));
match(resp, hash(pathtosz));

macro S = hash(k(Signer,Sanitizer));
macro L = hash(k(Signer, Verifier));
macro Sz = hash(hash(k(Signer, Verifier)), pathtosz);
macro R = hash(hash(L, pathtosz), pathtor);
macro ms = hash(messagefix, pathtor, a, S);
macro fl = hash(messagefix, pathtor, L);
macro mr = hash(messagefix, pathtor, a, R);

// assure both parties are alive
claim(Signer, Running, Sanitizer, messagefix,
a, pathtor, ms, fl, mr, Sz);
send_1(Signer, Sanitizer, {messagefix, a, pathtor, ms,
fl, mr, Sz}k(Signer, Sanitizer));

//recv acknowledgement from Sanitizer
var Rsz;
recv_11(Sanitizer, Signer, {Rsz}k(Sanitizer, Signer));
match(Rsz, hash(R));

// tell Scyther to verify claim that keys are secret
```

```
48 claim_a01(Signer, Secret, k(Verifier, Signer));
49 claim_a111(Signer, Secret, k(Verifier, Sanitizer));
50
51 // tell Scyther to verify claims that all secrets are truely secret
52 claim_a02(Signer, Secret, S);
53 claim_a03(Signer, Secret, L);
54 claim_a04(Signer, Secret, Sz);
55 claim_a05(Signer, Secret, R);
56 claim_a06(Signer, Secret, ms);
57 claim_a07(Signer, Secret, fl);
58 claim_a08(Signer, Secret, mr);
59 claim_a09(Signer, Secret, Rsz);
60
61 // tell Scyther to verify that all parties where really involved
62 claim_a10(Signer, Alive, Verifier);
63 claim_a11(Signer, Alive, Sanitizer);
64 claim_a12(Signer, Niagree);
65 claim_a13(Signer, Nisynch);
66
67 // tell Scyther to verify that all messages were sent and received
68 // by authentic parties
69 claim_a20(Signer, Commit, Verifier, pathtosz,
70 hash(pathtosz, k(Signer, Verifier)));
71 claim_a21(Signer, Commit, Sanitizer ,messagefix, a, pathtor,
72 ms, fl, mr, Sz);
73
74 // tell Scyther to verify that real Signer was really involved during
75 // the protocol
76 claim_a31(Signer, Reachable);
77
78 }
79
80 role Sanitizer{
81 var messagefix;
82 var a;
83 var pathtor;
84 var pathtosz;
85
86 recv_1(Signer, Sanitizer, {messagefix, a, pathtor,
87 ms, fl, mr, Sz}k(Signer, Sanitizer));
88
89 match(ms, hash(messagefix, pathtor, a, hash(k(Signer,Sanitizer))));
90
91 macro R = hash(Sz, pathtor);
92
93 //acknowledge receiving of R
94 claim(Sanitizer, Running, Signer,
95 {hash(R)}k(Sanitizer, Signer));
96
97 send_11(Sanitizer, Signer, {hash(R)}k(Sanitizer, Signer));
98
99 fresh atilde : String;
100 macro mrtilde = hash(messagefix, pathtor, atilde, R);
101
102 send_2(Sanitizer, Verifier,
103 {messagefix, atilde, pathtor, fl, mrtilde}k(Sanitizer, Verifier));
104
105 //recv acknowledgement of correct message
106 var resp21;
```

```
recv_21(Verifier,Sanitizer, {resp21}k(Verifier, Sanitizer));
match(resp21, hash(mrtilde));

// tell Scyther to verify claim that keys are secret
claim_b01(Sanitizer, Secret, k(Signer, Sanitizer));
claim_b111(Sanitizer, Secret, k(Signer, Sanitizer));

// tell Scyther to verify claims that all secrets are truely secret
claim_b02(Sanitizer, Secret, a);
claim_b03(Sanitizer, Secret, pathtor);
claim_b04(Sanitizer, Secret, pathtosz);
claim_b05(Sanitizer, Secret, ms);
claim_b06(Sanitizer, Secret, R);
claim_b07(Sanitizer, Secret, mrtilde);
claim_b08(Sanitizer, Secret, atilde);
claim_b09(Sanitizer, Secret, resp21);

// tell Scyther to verify that all parties where really involved
claim_b10(Sanitizer, Alive, Verifier);
claim_b11(Sanitizer, Alive, Signer);
claim_b12(Sanitizer, Niagree);
claim_b13(Sanitizer, Nisynch);

// tell Scyther to verify that all messages were sent and received
// by authentic parties
claim_b20(Sanitizer, Commit, Signer,
{hash(R)}k(Sanitizer, Signer)) ;
claim_b21(Sanitizer, Commit, Verifier,
{messagefix, atilde, pathtor, fl, mrtilde}k(Sanitizer, Verifier));

// tell Scyther to verify that real Sanitizer was really involved during
// involved during the protocol
claim_b31(Sanitizer, Reachable);

}

role Verifier{
var pathtosz;

recv_0(Signer, Verifier, {pathtosz,
hash(pathtosz, k(Signer, Verifier))}k(Signer, Verifier));
claim_c01(Verifier, Running,Signer,
{hash(pathtosz)}k(Verifier, Signer));
send_01(Verifier, Signer,
{hash(pathtosz)}k(Verifier, Signer));

var messagefix;
var pathtor;
var atilde;
//var mrtilde;

recv_2(Sanitizer, Verifier,
{messagefix, atilde, pathtor, fl, mrtilde}k(Sanitizer, Verifier));
match(mrtilde, hash(messagefix, pathtor, atilde,
hash(hash(k(Signer, Verifier)), pathtosz), pathtor));

//acknowledge everything is fine
claim_c02(Verifier, Running ,Sanitizer,
{hash(mrtilde)}k(Verifier, Sanitizer));
```

```
166 send_21(Verifier,Sanitizer, {hash(mrtilde)}k(Verifier, Sanitizer));
167
168 // tell Scyther to verify claim that keys are secret
169 claim_c04(Verifier, Secret, k(Signer, Verifier));
170 claim_c05(Verifier, Secret, k( Verifier, Signer));
171 claim_c06(Verifier, Secret, k( Verifier, Sanitizer));
172 claim_c07(Verifier, Secret, k( Sanitizer, Verifier));
173
174 // tell Scyther to verify claims that all secrets are truely secret
175 claim_c03(Verifier, Secret, pathtosz);
176 claim_c08(Verifier, Secret, pathtor);
177 claim_c09(Verifier, Secret, atilde);
178 claim_c09(Verifier, Secret, mrtilde);
179
180 // tell Scyther to verify that all parties where really involved
181 claim_c10(Verifier, Alive, Sanitizer);
182 claim_c11(Verifier, Alive, Signer);
183 claim_c12(Verifier, Niagree);
184 claim_c13(Verifier, Nisynch);
185
186 // tell Scyther to verify that all messages were sent and received
187 // by authentic parties
188 claim_c20(Verifier, Commit, Signer,
189 {hash(pathtosz)}k(Verifier, Signer));
190 claim_c21(Verifier, Commit, Verifier,
191 {hash(mrtilde)}k(Verifier, Sanitizer));
192
193 // tell Scyther to verify that real Verifier was really involved during
194 // the protocol
195 claim_c31(Verifier, Reachable);
196
197 }
198 }
```

The Scyther output can be seen in B.4.

- Claim that all keys are secret.
- Claim that all secrets are secret.
- Claim that all parties where really involved.
- Claim that all messages were sent and received by authentic parties.
- Claim that all roles were really and adequately involved during the protocol until the end.

Scyther did not found any attacks, i.e. deviations from the aforementioned claims, on the protocol under the assumptions that the protocol verification is decidable within 5 rounds and 10 patters per claim.

Scyther results : autoverify

Claim				Status	Comments
MallMACClaims	Signer	MallMACClaims,Signer2	Secret _Hidden_2	Ok	No attacks within bounds.
		MallMACClaims,Signer3	Secret _Hidden_1	Ok	No attacks within bounds.
		MallMACClaims,Signer4	Secret pathtosz	Ok	No attacks within bounds.
		MallMACClaims,Signer5	Secret pathtor	Ok	No attacks within bounds.
		MallMACClaims,Signer6	Secret a	Ok	No attacks within bounds.
		MallMACClaims,Signer7	Secret messagefix	Ok	No attacks within bounds.
		MallMACClaims,Signer8	Secret Rsz	Ok	No attacks within bounds.
		MallMACClaims,Signer9	Secret resp	Ok	No attacks within bounds.
		MallMACClaims,Signer10	Alive	Ok	No attacks within bounds.
		MallMACClaims,Signer11	Weakagree	Ok	No attacks within bounds.
		MallMACClaims,Signer12	Niagree	Ok	No attacks within bounds.
		MallMACClaims,Signer13	Nisynch	Ok	No attacks within bounds.
	Sanitizer	MallMACClaims,Sanitizer2	Secret _Hidden_4	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer3	Secret atilde	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer4	Secret _Hidden_3	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer5	Secret resp21	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer6	Secret pathtosz	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer7	Secret pathtor	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer8	Secret a	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer9	Secret messagefix	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer10	Alive	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer11	Weakagree	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer12	Niagree	Ok	No attacks within bounds.
		MallMACClaims,Sanitizer13	Nisynch	Ok	No attacks within bounds.
	Verifier	MallMACClaims,Verifier1	Secret _Hidden_5	Ok	No attacks within bounds.
		MallMACClaims,Verifier2	Secret atilde	Ok	No attacks within bounds.
		MallMACClaims,Verifier3	Secret pathtor	Ok	No attacks within bounds.
		MallMACClaims,Verifier4	Secret messagefix	Ok	No attacks within bounds.
		MallMACClaims,Verifier5	Secret pathtosz	Ok	No attacks within bounds.
		MallMACClaims,Verifier6	Alive	Ok	No attacks within bounds.
		MallMACClaims,Verifier7	Weakagree	Ok	No attacks within bounds.
		MallMACClaims,Verifier8	Niagree	Ok	No attacks within bounds.
		MallMACClaims,Verifier9	Nisynch	Ok	No attacks within bounds.

Done.

Figure B.4: Verification of the MallMACs protocol

B.4 Verification of the GroupMACs Protocol

For the verification of the GroupMACs protocol, Syther linux version 1.13 was used on an Ubuntu 64Bit 17.04 machine with Python version 2.7.13. Note that the Scyther folder has to be assigned execution rights for all files. The Scyther configuration was *typed matching*, *5 protocol runs as maximum bound*, *10 maximum number of patterns per claim* and *find all attacks*.

The Scyther model file of the protocol MallMACs is as follows:

```
hashfunction hash;

usertype String;
var time: Nonce;

protocol GroupMAC(Signer, Verifier, HP)
{

role Signer
{
fresh message: String;

macro K = hash(Signer, HP, k(Signer, HP));
macro h1 = hash(k(Signer, HP));
macro h2 = hash(h1);
claim(Signer, Running, Verifier, hash(message, h1), hash(message, K));
send_0(Signer, Verifier,
{message, h2, hash(message, h1), hash(message, K)}k(Signer, Verifier));

var mh1;
recv_3(Verifier, Signer, {hash(mh1)}k(Verifier, Signer));
match(hash(mh1), hash(hash(message, h1)));

// tell Scyther to verify claim that keys are secret
claim_a01(Signer, Secret, k(Signer, HP));

// tell Scyther to verify claims that all secrets are truely secret
claim_a02(Signer, Secret, K);
claim_a03(Signer, Secret, h1);
claim_a04(Signer, Secret, hash(mh1));

// tell Scyther to verify that all parties where really involved
claim_a10(Signer, Alive, Verifier);
claim_a11(Signer, Niagree);
claim_a15(Signer, Nisynch);

// tell Scyther to verify that all messages were sent and received
// by authentic parties
claim_a20(Signer, Commit, Verifier, hash(message, h1), hash(message, K));

// tell Scyther to verify that real Signer was really involved during
// the protocol
claim_a31(Signer, Reachable);

}

role Verifier
```

```
48 {
49 var message;
50
51 var mh1;
52 var mK;
53
54 recv_0(Signer, Verifier, {message, h2, mh1, mK}k(Signer, Verifier));
55 claim(Verifier, Running, HP, h2);
56 send_1(Verifier, HP, {h2}k(Verifier, HP));
57
58 recv_2(HP, Verifier, {h1, time}k(HP, Verifier));
59
60 match(mh1, hash(message, hash(mh1)));
61
62 claim(Verifier, Running, HP, hash(mh1));
63 send_3(Verifier, Signer, {hash(mh1)}k(Verifier, Signer));
64
65 claim(Verifier, Running, HP, hash(h1, time));
66 send_4(Verifier, HP, {hash(h1, time)}k(Verifier, Signer));
67
68
69 // tell Scyther to verify claims that all secrets are truely secret
70 claim_v00(Verifier, Secret, mh1);
71 claim_v01(Verifier, Secret, mK);
72 claim_v02(Verifier, Secret, h1);
73
74 // tell Scyther to verify claim that keys are secret
75 claim_v04(Verifier, Secret, k(Signer, Verifier));
76 claim_v04(Verifier, Secret, k(Verifier, HP));
77
78 // tell Scyther to verify that all parties where really involved
79 claim_v10(Verifier, Alive, Signer);
80 claim_v11(Verifier, Alive, HP);
81
82 claim_v12(Verifier, Weakagree, Signer);
83 claim_v13(Verifier, Weakagree, HP);
84
85 claim_v14(Verifier, Niagree);
86 claim_v15(Verifier, Nisynch);
87
88 // tell Scyther to verify that all messages were sent and received
89 // by authentic parties
90 claim_v22(Verifier, Commit, Signer, hash(mh1));
91 claim_v23(Verifier, Commit, HP, hash(h1, time));
92
93 // tell Scyther to verify that real Verifier was really involved during
94 // involved during the protocol
95 claim_v31(Verifier, Reachable);
96
97 }
98
99 role HP
100 {
101
102 recv_1(Verifier, HP, {h2}k(Verifier, HP));
103
104 match(h2, hash(hash(k(Signer, HP))));
105
106 fresh time: Nonce;
```

```
107
108 claim(HP, Running, Verifier, h1);
109 send_2(HP, Verifier, {h1, time}k(HP, Verifier));
110
111 var resp;
112 recv_4(Verifier, HP, {resp}k(Verifier, Signer));
113 match(hash(h1, time), resp);
114
115 // tell Scyther to verify claim that keys are secret
116 claim_h00(HP, Secret, k(Verifier, Signer));
117 claim_h03(HP, Secret, k(HP, Verifier));
118 claim_h04(HP, Secret, k(Signer, HP));
119
120 // tell Scyther to verify claims that all secrets are truely secret
121 claim_h01(HP, Secret, hash(k(Signer, HP)));
122 claim_h02(HP, Secret, h1);
123
124
125 // tell Scyther to verify that all parties where really involved
126 claim_h11(HP, Alive, HP);
127
128 claim_h13(HP, Weakagree, HP);
129
130 claim_h14(HP, Niagree);
131 claim_h15(HP, Nisynch);
132
133
134 // tell Scyther to verify that all messages were sent and received
135 // by authentic parties
136 claim_h23(HP, Commit, Verifier, h1);
137
138 // tell Scyther to verify that real HP was really involved during
139 // the protocol
140 claim_h31(HP, Reachable);
141 }
142 }
```

The Scyther output can be seen in B.5.

Following claims have been defined (for specifics, please refer to the protocol description):

- Claim that all keys are secret.
- Claim that all secrets are secret.
- Claim that all parties where really involved.
- Claim that all messages were sent and received by authentic parties.
- Claim that all roles were really and adequately involved during the protocol until the end.

Scyther did not found any attacks, i.e. deviations from the aforementioned claims, on the protocol under the assumptions that the protocol verification is decidable within 5 rounds and 10 patters per claim.

Scyther results : autoverify

Claim				Status	Comments
GroupMAC	Signer	GroupMAC,Signer2	Secret _Hidden_ 1	Ok	No attacks within bounds.
		GroupMAC,Signer3	Secret message	Ok	No attacks within bounds.
		GroupMAC,Signer4	Secret mh1	Ok	No attacks within bounds.
		GroupMAC,Signer5	Alive	Ok	No attacks within bounds.
		GroupMAC,Signer6	Weakagree	Ok	No attacks within bounds.
		GroupMAC,Signer7	Niagree	Ok	No attacks within bounds.
		GroupMAC,Signer8	Nisynch	Ok	No attacks within bounds.
	Verifier	GroupMAC,Verifier2	Secret _Hidden_ 2	Ok	No attacks within bounds.
		GroupMAC,Verifier3	Secret mK	Ok	No attacks within bounds.
		GroupMAC,Verifier4	Secret mh1	Ok	No attacks within bounds.
		GroupMAC,Verifier5	Secret message	Ok	No attacks within bounds.
		GroupMAC,Verifier6	Alive	Ok	No attacks within bounds.
		GroupMAC,Verifier7	Weakagree	Ok	No attacks within bounds.
		GroupMAC,Verifier8	Niagree	Ok	No attacks within bounds.
		GroupMAC,Verifier9	Nisynch	Ok	No attacks within bounds.
	HP	GroupMAC,HP2	Secret _Hidden_ 4	Ok	No attacks within bounds.
		GroupMAC,HP3	Secret time	Ok	No attacks within bounds.
		GroupMAC,HP4	Secret _Hidden_ 3	Ok	No attacks within bounds.
		GroupMAC,HP5	Secret resp	Ok	No attacks within bounds.
		GroupMAC,HP6	Alive	Ok	No attacks within bounds.
		GroupMAC,HP7	Weakagree	Ok	No attacks within bounds.
		GroupMAC,HP8	Niagree	Ok	No attacks within bounds.
		GroupMAC,HP9	Nisynch	Ok	No attacks within bounds.

Done.

Figure B.5: Verification of the GroupMACs protocol

Bibliography

[AAC16] Aziz, Benjamin (Hrsg.) ; Arenas, Alvaro (Hrsg.) ; Crispo, Bruno (Hrsg.): *Engineering Secure Internet of Things Systems*. Institution of Engineering and Technology, 2016 (Sector Publications). http://digital-library.theiet.org/content/books/se/pbse002e

[AB09] Agrawal, Shweta ; Boneh, Dan: Homomorphic MACs: MAC-based integrity for network coding. In: *Applied Cryptography and Network Security* Springer, 2009, S. 292–305

[ACDMT05] Ateniese, Giuseppe ; Chou, Daniel H. ; De Medeiros, Breno ; Tsudik, Gene: Sanitizable signatures. In: *Computer Security–ESORICS 2005*. Springer, 2005, S. 159–177

[Act10] Action, Buzz C.: *Google Buzz Class Action*. Archived on webcitation.org. http://www.webcitation.org/5tyF08T40. Version: 12 2010. – Last accessed on August 17, 2017

[AFHPW15] Angulo, Julio ; Fischer-Hübner, Simone ; Pulls, Tobias ; Wästlund, Erik: Usable transparency with the data track: a tool for visualizing data disclosures. In: *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems* ACM, 2015, S. 1803–1808

[AIM10a] Atzori, Luigi ; Iera, Antonio ; Morabito, Giacomo: The Internet of Things: A survey. In: *Computer Networks* 54 (2010), Nr. 15, 2787 - 2805. http://dx.doi.org/http://dx.doi.org/10.1016/j.comnet.2010.05.010. – DOI http://dx.doi.org/10.1016/j.comnet.2010.05.010. – ISSN 1389–1286

[AIM10b] Atzori, Luigi ; Iera, Antonio ; Morabito, Giacomo: The internet of things: A survey. In: *Computer networks* 54 (2010), Nr. 15, S. 2787–2805

[AIM14] Atzori, Luigi ; Iera, Antonio ; Morabito, Giacomo: From" smart objects" to" social objects": The next evolutionary step of the

internet of things. In: *IEEE Communications Magazine* 52 (2014), Nr. 1, S. 97–105

[ALM87] Appelbaum, Paul S. ; Lidz, Charles W. ; Meisel, Alan: *Informed consent: Legal theory and clinical practice*. JSTOR, 1987

[Ama14] Amazon.com, Inc.: *Nike+ Stand Alone Sensor Kit*. Electronic. http://www.amazon.com/Nike-Stand-Alone-Sensor-Kit/dp/B001L6LJJS/ref=sr_1_3?ie=UTF8&qid=1462287065&sr=8-3&keywords=nike%2B. Version: May 2014. – Price - $11.22, last checked on August 17, 2017.

[Ama16a] Amazon: *AWS IoT*. Electronic. https://aws.amazon.com/iot/how-it-works/?nc1=h_ls. Version: 2016. – lasti visited August 17, 2017

[Ama16b] Amazon.com, Inc.: *Nike Men's Air Zoom Pegasus 32 Running Shoe, Nike+ Ready*. Electronic. http://www.amazon.com/s/ref=nb_sb_noss_2?url=search-alias%3Daps&field-keywords=nike+running&rh=i%3Aaps%2Ck%3Anike+running. Version: May 2016. – Price - $71,00, last checked on August 17, 2017.

[Ama16c] Amazon.com, Inc.: *Raspberry Pi 3 Model B*. Electronic. http://www.amazon.com/Raspberry-Pi-RASP-PI-3-Model-Motherboard/dp/B01CD5VC92/ref=sr_1_1?ie=UTF8&qid=1462545250&sr=8-1&keywords=raspberry. Version: May 2016. – Price - $40,00, last checked on August 17, 2017.

[ANN06] Abdalla, Michel ; Namprempre, Chanathip ; Neven, Gregory: On the (im) possibility of blind message authentication codes. In: *Topics in Cryptology–CT-RSA 2006*. Springer, 2006, S. 262–279

[Ash09] Ashton, Kevin: That 'internet of things' thing. In: *RFiD Journal* 22 (2009), Nr. 7, S. 97–114

[Ass11] Association, IEEE S.: *IEEE 802.15: Wireless Personal Area Networks (PANs)*. Electronic. http://standards.ieee.org/about/get/802/802.15.html. Version: 2003-2011

[B+01] Bisdikian, Chatschik u. a.: An overview of the Bluetooth wireless technology. In: *IEEE Commun Mag* 39 (2001), Nr. 12, S. 86–94

[BBF+13] Bassi, Alessandro (Hrsg.) ; Bauer, Martin (Hrsg.) ; Fiedler, Martin (Hrsg.) ; Kramp, Thorsten (Hrsg.) ; Kranenburg, Rob v.

(Hrsg.) ; Lange, Sebastian (Hrsg.) ; Meissner, Stefan (Hrsg.): *Enabling Things to Talk: Designing IoT solutions with the IoT Architectural Reference Model*. Berlin : Springer, 2013. – ISBN 978–3–642–40402–3

[BBS04] Boneh, Dan ; Boyen, Xavier ; Shacham, Hovav: Short group signatures. In: *Advances in Cryptology–CRYPTO 2004* Springer, 2004, S. 41–55

[BEK14] Bormann, Carsten ; Ersue, Mehmet ; Keranen, A: Terminology for Constrained-Node Networks. 2014. – Forschungsbericht

[Ben14] Bence, Steve: *The cost breakdown of a $100 pair of sneakers*. Electronic. http://www.bizjournals.com/portland/blog/threads_and_laces/2014/12/the-cost-breakdown-of-a-100-pair-of-sneakers.html. Version: December 2014

[Ber08] Bernstein, Daniel J.: ChaCha, a variant of Salsa20. In: *Workshop Record of SASC* Bd. 8, 2008

[BFH13] Beckers, Kristian ; Fassbender, Stephan ; Heisel, Maritta: A Meta-Model Approach to the Fundamentals for a Pattern Language for Context Elicitation. In: *Proceedings of the 18th European Conference on Pattern Languages of Programs (Europlop)*, ACM, 2013, S. –. – Accepted for Publication

[BHWW09] Buttyán, Levente ; Holczer, Tamás ; Weimerskirch, André ; Whyte, William: Slow: A practical pseudonym changing scheme for location privacy in vanets. In: *Vehicular Networking Conference (VNC), 2009 IEEE* IEEE, 2009, S. 1–8

[BJS07] Barker, Elaine ; Johnson, Don ; Smid, Miles: NIST special publication 800-56A: Recommendation for pair-wise key establishment schemes using discrete logarithm cryptography (revised). In: *Computer Security, National Institute of Standards and Technology (NIST), published by NIST* (2007)

[BKB16] Bier, Christoph ; Kühne, Kay ; Beyerer, Jürgen: PrivacyInsight: The Next Generation Privacy Dashboard. In: *Annual Privacy Forum* Springer, 2016, S. 135–152

[BKS14] Busch, Marianne ; Koch, Nora ; Suppan, Santiago: Modeling Security Features of Web Applications. Version: 2014. http://dx.

doi.org/10.1007/978-3-319-07452-8_5. In: Heisel, Maritta (Hrsg.) ; Joosen, Wouter (Hrsg.) ; Lopez, Javier (Hrsg.) ; Martinelli, Fabio (Hrsg.): *Engineering Secure Future Internet Services and Systems* Bd. 8431. Springer International Publishing, 2014. – ISBN 978–3–319–07451–1, 119-139

[BLP+13] Bilimoria, Karl Y. ; Liu, Yaoming ; Paruch, Jennifer L. ; Zhou, Lynn ; Kmiecik, Thomas E. ; Ko, Clifford Y. ; Cohen, Mark E.: Development and evaluation of the universal ACS NSQIP surgical risk calculator: a decision aid and informed consent tool for patients and surgeons. In: *Journal of the American College of Surgeons* 217 (2013), Nr. 5, S. 833–842

[Bor15] Bormann, Carsten: *Ten years of standardizing the "Internet of Things" in the IETF*. http://www.w3.org/2015/04/munich/bormann.pdf. Version: 2015

[BPC+07] Baronti, Paolo ; Pillai, Prashant ; Chook, Vince W. ; Chessa, Stefano ; Gotta, Alberto ; Hu, Y F.: Wireless sensor networks: A survey on the state of the art and the 802.15. 4 and ZigBee standards. In: *Computer communications* 30 (2007), Nr. 7, S. 1655–1695

[BS93] Bellotti, Victoria ; Sellen, Abigail: Design for privacy in ubiquitous computing environments. In: *Proceedings of the Third European Conference on Computer-Supported Cooperative Work 13–17 September 1993, Milan, Italy ECSCW'93* Springer, 1993, S. 77–92

[BS04] Beresford, Alastair R. ; Stajano, Frank: Mix zones: User privacy in location-aware services. (2004)

[Bun14] Bundesministerium für Bildung und Forschung: *Industrie 4.0 - Innovationen für die Produktion von morgen*. Electronic. https://www.bmbf.de/pub/Industrie_4.0.pdf. Version: 2014. – lasti visited August 17, 2017

[BW15] Buttarelli, Giovanni ; Wiewiorowski, Wojciech: *The European Data Protection Supervisor Data Protection Glossary*. Web Page. https://secure.edps.europa.eu/EDPSWEB/edps/EDPS/Dataprotection/Glossary. Version: July 2015. – (accessed 2015/07/14)

[C+12a] Commission, EC E. u. a.: Proposal for a Regulation of the European Parliament and of the Council on the Protection of Indi-

viduals with Regard to the Processing of Personal Data and on the Free movement of Such Data (General Data Protection Regulation). In: *COM (2012) 11 final, 2012/0011 (COD), Brussels, 25 January 2012* (2012). http://statewatch.org/news/2015/dec/eu-council-dp-reg-draft-final-compromise-15039-15.pdf

[C+12b] Commission, EU u. a.: Proposal for a Regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation)(Vol. 11). In: *ec. europa. eu/justice/dataprotection/document/review2012/com_2012_11_en. pdf* (2012)

[Can] Canada, Office of the Privacy Comissioner o.: *Key Steps for Organizations in Responding to Privacy Breaches*. https://www.priv.gc.ca/information/guide/2007/gl_070801_02_e.asp. – last accessed on August 17, 2017

[Car96] Carpenter, Brian E.: Architectural principles of the Internet. (1996)

[Cav09a] Cavoukian, Ann: *7 Foundational Principles for Privacy By Design*. Web Page. https://www.privacybydesign.ca/index.php/about-pbd/7-foundational-principles. Version: January 2009. – (accessed 2015/07/14), Canada

[Cav09b] Cavoukian, Ann: Privacy by design: The 7 foundational principles. In: *Information and Privacy Commissioner of Ontario, Canada* (2009)

[cc2] *CC2530 Second Generation System-on-Chip Solution for 2.4 GHz IEEE 802.15.4 / RF4CE / ZigBee*. – Available online at http://www.ti.com/tool/cc3100boost, last accessed on 2015/11/26.

[CD12] Chandra, P. ; Deleersnyder, S.: *OWASP Software Assurance Maturity Model*. 2012

[Cer] Cervesato, Iliano: The Dolev-Yao intruder is the most powerful attacker

[CG15] Carrera, Sergio ; Guild, Elspeth: Safe Harbour or into the storm? EU-US data transfers after the Schrems judgment. Liberty and security in Europe No. 85/November 2015. (2015)

[Che15] Chen, Wenwen: *An Authentication and Authorization Protocol for Constrained Devices*. Master Thesis, December 2015

[CJ11] Cheng, Chi ; Jiang, Tao: A Novel Homomorphic MAC Scheme for Authentication in Network Coding. In: *Communications Letters, IEEE* 15 (2011), November, Nr. 11, S. 1228–1230. http://dx.doi.org/10.1109/LCOMM.2011.090911.111531. – DOI 10.1109/LCOMM.2011.090911.111531. – ISSN 1089–7798

[CK09] Cao, Lili ; Krumm, John: From GPS traces to a routable road map. In: *Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems* ACM, 2009, S. 3–12

[CKLM13] Chase, Melissa ; Kohlweiss, Markulf ; Lysyanskaya, Anna ; Meiklejohn, Sarah: Malleable Signatures: Complex Unary Transformations and Delegatable Anonymous Credentials. In: *IACR Cryptology ePrint Archive* 2013 (2013), S. 179

[Cla02] Clarke, Siobhán: Extending standard UML with model composition semantics. In: *Science of Computer Programming* 44 (2002), Nr. 1, S. 71–100

[CLN09] In: Cremers, Cas J. F. ; Lafourcade, Pascal ; Nadeau, Philippe: *Comparing State Spaces in Automatic Security Protocol Analysis*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2009. – ISBN 978–3–642–02002–5, 70–94

[CM05] In: Cremers, Cas ; Mauw, Sjouke: *Operational Semantics of Security Protocols*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2005. – ISBN 978–3–540–32032–6, 66–89

[CML06] Chow, Chi-Yin ; Mokbel, Mohamed F. ; Liu, Xuan: A peer-to-peer spatial cloaking algorithm for anonymous location-based service. In: *Proceedings of the 14th annual ACM international symposium on Advances in geographic information systems* ACM, 2006, S. 171–178

[CMV02] Chandra, Pravir ; Messier, Matt ; Viega, John: Network security with OpenSSL. In: *O'Reily, June* 1 (2002), S. 44–68

[Com15a] Commission, EC E.: Agreement on Commission's EU data protection reform will boost Digital Single Market. In: *European Commission - Press release* (2015), December. http://statewatch.org/news/2015/dec/eu-council-dp-reg-draft-final-compromise-15039-15.pdf

[Com15b] Company, McKinsey ⋮ *THE INTERNET OF THINGS: MAPPING THE VALUE BEYOND THE HYPE*. Electronic. http://www.mckinsey.com/~/media/McKinsey/Business%20Functions/Business%20Technology/Our%20Insights/The%20Internet%20of%20Things%20The%20value%20of%20digitizing%20the%20physical%20world/Unlocking_the_potential_of_the_Internet_of_Things_Full_report.ashx. Version: June 2015

[Con02] Consortium, World Wide W.: A P3P Preference Exchange Language 1.0 (APPEL1. 0). In: *W3C Working Draft, April* (2002). https://www.w3.org/TR/P3P-preferences/

[Cor12a] Corporation, Microsoft: *Microsoft Security Development Lifecycle (SDL) – Process Guidance*. Electronic. https://msdn.microsoft.com/de-de/library/windows/desktop/84aed186-1d75-4366-8e61-8d258746bopq.aspx. Version: 2012. – last visited on August 17, 2017.

[Cor12b] Corporation, Microsoft ; Microsoft Corporation (Hrsg.): *Security Development Lifecycle - SDL Process Guidance Version 5.2*. 5.2-1. Microsoft Corporation, May 2012. http://www.microsoft.com/en-us/download/confirmation.aspx?id=29884

[Cor14] Cornish, Audie: *How Much Will The Hack Cost Sony?* Electronic, Audio. http://www.npr.org/2014/12/18/371721061/how-much-will-the-hack-cost-sony. Version: December 2014

[Cor16] Cornell Cornell University Law School Search: *Jurisprudence*. Electronic. https://www.law.cornell.edu/wex/jurisprudence. Version: 2016. – last visited on August 17, 2017

[Cou08] Council, NI: Disruptive civil technologies: Six technologies with potential impacts on us interests out to 2025. In: *Conference Report CR* Bd. 2007, 2008

[Cra02] Cranor, Lorrie: *Web privacy with P3P*. " O'Reilly Media, Inc.", 2002

[Cra03] Cranor, L. F.: P3P: making privacy policies more useful. In: *IEEE Security Privacy* 1 (2003), Nov, Nr. 6, S. 50–55. http://dx.doi.org/10.1109/MSECP.2003.1253568. – DOI 10.1109/MSECP.2003.1253568. – ISSN 1540–7993

[Cra09] Cranor, Lorrie: *IE Privacy Bird: Find web sites that respect your privacy*. Web Page. http://www.privacybird.org/. Version: 2009. – (accessed 2015/07/16)

[Cre08] In: Cremers, Cas J. F.: *The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols*. Berlin, Heidelberg : Springer Berlin Heidelberg, 2008. – ISBN 978–3–540–70545–1, 414–418

[Cre14a] Cremers, Cas: *Scyther*. Electronic, April 2014. – Available online at http://www.cs.ox.ac.uk/people/cas.cremers/scyther/index.html, last accessed on 2015/11/10.

[Cre14b] Cremers, Cas: *Scyther User Manual*. Februar 2014 Available online at https://github.com/cascremers/scyther/blob/master/gui/scyther-manual.pdf, last accessed on 2015/11/10.

[CSH15] Cuellar, Jorge ; Suppan, Santiago ; Henrich, Poehls: *Internet Draft: Privacy-Enhanced Tokens for Authorization in ACE*. 2015

[CSP15] Cuellar, J. ; Suppan, S. ; Poehls, H. C.: *Privacy-Enhanced Tokens for Authorization in ACE*. October 2015

[CSPH12] Costante, Elisa ; Sun, Yuanhao ; Petković, Milan ; Hartog, Jerry den: A machine learning solution to assess privacy policy completeness:(short paper). In: *Proceedings of the 2012 ACM workshop on Privacy in the electronic society* ACM, 2012, S. 91–96

[Cuk14] Cukier, Kenneth: *Big data is better data*. https://www.ted.com/talks/kenneth_cukier_big_data_is_better_data?language=en. Version: 2014

[CVH91] Chaum, David ; Van Heyst, Eugene: Group signatures. In: *Advances in Cryptology EUROCRYPT 91* Springer, 1991, S. 257–265

[CW07] Cranor, Lorrie ; Wenning, Rigo: *Platform for Privacy Preferences (P3P) Project*. Web Page. http://www.w3.org/P3P/. Version: November 2007. – (accessed 2015/07/14)

[CZSSM80] Cassileth, Barrie R. ; Zupkis, Robert V. ; Sutton-Smith, Katherine ; March, Vicki: Informed consent—why are its goals imperfectly realized? In: *New England journal of medicine* 302 (1980), Nr. 16, S. 896–900

[DBNA14] Daugherty, Paul ; Banerjee, Prith ; Negm, Walid ; Alte, Allan E.: *Driving Unconventional Growth through the Industrial Internet of Things*. Electronic. https://www.accenture.com/t20150523T120435___w___/us-en/_acnmedia/Accenture/Conversion-Assets/DotCom/Documents/Global/PDF/Dualpub_12/Accenture-Driving-Unconventional-Growth-through-IIoT.pdf. Version: 2014. – last visited August 17, 2017

[Dea15] Dean, Benjamin: *Why companies have little incentive to invest in cybersecurity*. Electronic. http://theconversation.com/why-companies-have-little-incentive-to-invest-in-cybersecurity-37570. Version: March 2015

[DeB15] DeBord, Matthew: *Connected cars are almost here*. Electronic. http://www.businessinsider.com/connected-cars-2015-9?IR=T. Version: September 2015

[Dev11] Devices, Memory P.: *CR 2032 - Data Sheets & Articles for Designing with CR2032 Batteries*. http://cr2032.co/. Version: 2011. – last accessed on August 17, 2017

[DFF14] Dennedy, Michelle F. ; Fox, Jonathan ; Finneran, Thomas R.: *The Privacy Engineer's Manifesto: Getting from Policy to Code to QA to Value*. 1st. Apress, 2014

[DG12] Diaz, Claudia ; Gürses, Seda: Understanding the landscape of privacy technologies. In: *Proc. of the Information Security Summit* (2012), S. 58–63

[DGZM15] Doty, Nick P. ; Gupta, Mohit ; Zych, Jeff ; McDonald, Rowyn: *Privacy dashboard, a privacy pattern*. Web Page. http://privacypatterns.org/patterns/Privacy-dashboard. Version: July 2015. – (accessed 2015/07/22), University of Berkley, School of Information

[DH15] Das, Raghu ; Harrop, Peter: *RFID Forecasts, Players and Opportunities 2016-2026*. Electronic. http://www.idtechex.com/research/reports/rfid-forecasts-players-and-opportunities-2016-2026-000451.asp. Version: October 2015. – lasti visited August 17, 2017

[Dif88] Diffie, Whitfield: The first ten years of public-key cryptography. In: *Proceedings of the IEEE* 76 (1988), Nr. 5, S. 560–577

[Dir95] Directive, EU: 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. In: *Official Journal of the EC* 23 (1995), Nr. 6

[DL01] Delucchi, Mark A. ; Lipman, Timothy E.: An analysis of the retail and lifecycle cost of battery-powered electric vehicles. In: *Transportation Research Part D: Transport and Environment* 6 (2001), Nr. 6, S. 371–404

[Dob12] Dobkin, Daniel M.: *The RF in RFID: UHF RFID in practice*. Newnes, 2012. – ISBN 978–0–12–394583–9

[DR98] Dugatkin, Lee A. ; Reeve, Hudson K.: *Game theory and animal behavior*. Oxford University Press, 1998

[Dun06] Dunn, M.W.: *User-centric consent management system and method*. https://www.google.com/patents/US7076558. Version: Juli 11 2006. – US Patent 7,076,558

[DW10] Doty, Nick ; Wilde, Erik: Geolocation privacy and application platforms. In: *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Security and Privacy in GIS and LBS* ACM, 2010, S. 65–69

[DY83] Dolev, D. ; Yao, A.: On the security of public key protocols. In: *IEEE Transactions on Information Theory* 29 (1983), Mar, Nr. 2, S. 198–208. http://dx.doi.org/10.1109/TIT.1983.1056650. – DOI 10.1109/TIT.1983.1056650. – ISSN 0018–9448

[ECJ02] Elias, Arun A. ; Cavana, Robert Y. ; Jackson, Laurie S.: Stakeholder analysis for R&D project management. In: *R&D Management* 32 (2002), Nr. 4, S. 301–310

[ECoD99] Economic Co-operation, Organisation for ; Development: *OECD principles of corporate governance*. OECD, 1999

[EU 14] EU Article 29 Data Protection Working Party: Opinion 8/2014 on the Recent Developments on the Internet of Things / European Union. Version: September 2014. http://ec.europa.eu/justice/data-protection/article-29/documentation/opinion-recommendation/files/2014/wp223_en.pdf. 2014 (14/EN WP 223). – Forschungsbericht. – (accessed 2015/07/14)

[EU14] European Union, Council of t.: Council Conclusions on future networks and the internet. In: *2907th TRANSPORT, TELECOMMUNICATIONS and ENERGY Council meeting*, 2014

[Eur08] European Parliament, Council of the European Union: *Directive 2008/50/EC of the European Parliament and of the Council of 21 May 2008 on ambient air quality and cleaner air for Europe.* Electronic. http://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=CELEX:32008L0050:en:NOT. Version: June 2008. – last visited on August 17, 2017

[FDW04] Feldhofer, Martin ; Dominikus, Sandra ; Wolkerstorfer, Johannes: Strong authentication for RFID systems using the AES algorithm. In: *International Workshop on Cryptographic Hardware and Embedded Systems* Springer, 2004, S. 357–370

[Fed07] Federrath, Hannes: JAP—Anonymity & Privacy. 10 (2007). http://anon.inf.tu-dresden.de. – last checked on August 17, 2017.

[FHAP13] Fischer-Hübner, Simone ; Angulo, Julio ; Pulls, Tobias: How can cloud users be supported in deciding on, tracking and controlling how their data are used? In: *IFIP PrimeLife International Summer School on Privacy and Identity Management for Life* Springer, 2013, S. 77–92

[FLM05] Friedman ; Lin ; Miller: Informed Consent by Design. In: *in Security and Usability*, 2005, 503-530. – (accessed 2015/07/14)

[FRE15] FREMANTLE, PAUL: A REFERENCE ARCHITECTURE FOR THE INTERNET OF THINGS. 0.9.0 (2015), S. 1–21

[G+00] Gambetta, Diego u. a.: Can we trust trust. In: *Trust: Making and breaking cooperative relations* 13 (2000), S. 213–237

[GBB14] Gerdes, Stefanie ; Bergmann, Olaf ; Bormann, Carsten: Delegated CoAP authentication and authorization framework (DCAF). In: *IETF draftgerdes-core-dcaf-authorize-02* (2014)

[GBMP13] Gubbi, Jayavardhana ; Buyya, Rajkumar ; Marusic, Slaven ; Palaniswami, Marimuthu: Internet of Things (IoT): A vision, architectural elements, and future directions. In: *Future Generation Computer Systems* 29 (2013), Nr. 7, S. 1645–1660

[GGM86] Goldreich, Oded ; Goldwasser, Shafi ; Micali, Silvio: How to construct random functions. In: *Journal of the ACM (JACM)* 33 (1986), Nr. 4, S. 792–807

[GKN+11] Gluhak, A. ; Krco, S. ; Nati, M. ; Pfisterer, D. ; Mitton, N. ; Razafindralambo, T.: A survey on facilities for experimental internet of things research. In: *Communications Magazine, IEEE* 49 (2011), November, Nr. 11, S. 58–67. http://dx.doi.org/10.1109/MCOM.2011.6069710. – DOI 10.1109/MCOM.2011.6069710. – ISSN 0163–6804

[GMKK+13] Garcia-Morchon, Oscar ; Keoh, Sye L. ; Kumar, Sandeep ; Moreno-Sanchez, Pedro ; Vidal-Meca, Francisco ; Ziegeldorf, Jan H.: Securing the IP-based Internet of Things with HIP and DTLS. In: *Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*. New York, NY, USA : ACM, 2013 (WiSec '13). – ISBN 978–1–4503–1998–0, 119–124

[GNC+01] Gutierrez, Jose A. ; Naeve, Marco ; Callaway, Ed ; Bourgeois, Monique ; Mitter, Vinay ; Heile, Bob: IEEE 802.15. 4: a developing standard for low-power low-cost wireless personal area networks. In: *network, IEEE* 15 (2001), Nr. 5, S. 12–19

[Gro10] Gross, Doug: *Google Buzz goes after Facebook, Twitter*. Electronic. http://edition.cnn.com/2010/TECH/02/09/google.social/. Version: February 2010

[GSM15] GSMA: *How China is scaling the Internet of Things*. Electronic. http://www.gsma.com/newsroom/wp-content/uploads/16531-China-IoT-Report-LR.pdf. Version: 2015. – lasti visited August 17, 2017

[GTD11] Gürses, Seda ; Troncoso, Carmela ; Diaz, Claudia: Engineering privacy by design. In: *Computers, Privacy & Data Protection* 14 (2011)

[Gür14] Gürses, Seda: Can you engineer privacy? In: *Communications of the ACM* 57 (2014), Nr. 8, S. 20–23

[GW13] Gennaro, Rosario ; Wichs, Daniel: Fully homomorphic message authenticators. In: *Advances in Cryptology-ASIACRYPT 2013*. Springer, 2013, S. 301–320

[Har12] Hardt, D.: The OAuth 2.0 Authorization Framework / Internet Engineering Task Force (IETF). Version: October 2012. http://tools.ietf.org/html/rfc6749. 2012 (RFC 6749). – Standard. – (accessed 2015/07/26)

[HB95] Hes, Ronald ; Borking, Johannes Josephus Franciscus M.: *Privacy-enhancing technologies: The path to anonymity.* Registratiekamer, 1995 https://www.ipc.on.ca/english/Resources/Discussion-Papers/Discussion-Papers-Summary/?id=329

[HBG+13] Hahm, Oliver ; Baccelli, Emmanuel ; Günes, Mesut ; Wählisch, Matthias ; Schmidt, Thomas C.: RIOT OS: Towards an OS for the Internet of Things. In: *Proceedings of the 32nd IEEE International Conference on Computer Communications (INFOCOM), Poster Session*, 2013

[HCFF08] Henzen, Luca ; Carbognani, Flavio ; Felber, Norbert ; Fichtner, Wolfgang: VLSI hardware evaluation of the stream ciphers Salsa20 and ChaCha, and the compression function Rumba. In: *Signals, Circuits and Systems, 2008. SCS 2008. 2nd International Conference on* IEEE, 2008, S. 1–5

[HCH+05] Hughes, John ; Cantor, Scott ; Hodges, Jeff ; Hirsch, Frederick ; Mishra, Prateek ; Philpott, Rob ; Maler, Eve: Profiles for the oasis security assertion markup language (saml) v2. 0. In: *OASIS standard* 15 (2005)

[Hil76] Hillman, Robert A.: Construction of the Uniform Commercial Code: UCC Section 1-103 and Code Methodology. In: *BC Indus. & Com. L. Rev.* 18 (1976), S. 655

[HL06] Howard, M. ; Lipner, S.: *The Security Development Lifecycle : SDL : A Process for Developing Demonstrably More Secure Software.* Microsoft Press, 2006

[HMPR04] Hevner, Alan ; March, Salvatore ; Park, Jinsoo ; Ram, Sudha: Design Science in Information Systems Research. In: *MIS Quarterly* 28 (2004), Nr. 1, S. 75–105

[HMV06] Hankerson, Darrel ; Menezes, Alfred J. ; Vanstone, Scott: *Guide to elliptic curve cryptography.* Springer Science & Business Media, 2006

[HMYS05] Huang, Leping ; Matsuura, Kanta ; Yamane, Hiroshi ; Sezaki, Kaoru: Enhancing wireless location privacy using silent period. In: *Wireless Communications and Networking Conference, 2005 IEEE* Bd. 2 IEEE, 2005, S. 1187–1192

[Hoe14] Hoepman, Jaap-Henk: Privacy design strategies. In: *ICT Systems Security and Privacy Protection*. Springer, 2014, S. 446–459

[HS05] Health, Department of ; Services, Human: *SELECTING A DEVELOPMENT APPROACH*. https://www.cms.gov/Research-Statistics-Data-and-Systems/CMS-Information-Technology/XLC/Downloads/SelectingDevelopmentApproach.pdf. Version: February 2005

[HTQ12] Haduong ; Tordillos ; Quintana: *Privacy Simplified Icons*. Web Page. http://yale.edu/self/psicons.html. Version: May 2012. – (accessed 2015/07/14)

[HW04] Hossain, Liaquat ; Wigand, Rolf T.: ICT enabled virtual collaboration through trust. In: *Journal of Computer-Mediated Communication* 10 (2004), Nr. 1, S. 00–00

[HWL04] Han, Song ; Wang, Jie ; Liu, Wanquan: An Efficient Identity-Based Group Signature Scheme over Elliptic Curves. In: Freire, MárioMarques (Hrsg.) ; Chemouil, Prosper (Hrsg.) ; Lorenz, Pascal (Hrsg.) ; Gravey, Annie (Hrsg.): *Universal Multiservice Networks* Bd. 3262. Springer Berlin Heidelberg, 2004. – ISBN 978–3–540–23551–4, S. 417–429

[HZH11] Holtz, Leif-Erik ; Zwingelberg, Harald ; Hansen, Marit: Privacy policy icons. In: *Privacy and Identity Management for Life*. Springer, 2011, S. 279–285

[IBM16] IBM: *Start developing with IBM Watson IoT Platform*. Electronic. http://www.ibm.com/analytics/us/en/internet-of-things/. Version: 2016. – lasti visited August 17, 2017

[ICT+13] Ishaq, Isam ; Carels, David ; Teklemariam, Girum K. ; Hoebeke, Jeroen ; Abeele, Floris Van d. ; Poorter, Eli D. ; Moerman, Ingrid ; Demeester, Piet: IETF standardization in the field of the internet of things (IoT): a survey. In: *Journal of Sensor and Actuator Networks* 2 (2013), Nr. 2, S. 235–287

[Inc16] Inc., Googe: *Google Dashboard*. Electronic. https://support.google.com/accounts/answer/162744?hl=en. Version: 2016. – last visited on August 17, 2017.

[Ins14] Instruments, Texas: Red Hat GCC for MSP430TM Microcontrollers / Texas Instruments. 2014 (SLAU591A). – Quick Start Guide. – Available at http://www.ti.com/lit/ml/slau591a/slau591a.pdf. Download at http://software-dl.ti.com/msp430/msp430_public_sw/mcu/msp430/MSPGCC/latest/index_FDS.html, last accessed on 2015/11/10.

[Ins15] Institute, German Federal Highway R.: *Testfeld zur Erprobung von Verkehrsdaten-Erfassungsgeräten*. Electronic. http://www.bast.de/DE/Verkehrstechnik/Fachthemen/v5-verkehrserfassung/Testfeld-A4.html?nn=605156. Version: 2015. – last visited on August 17, 2017

[Int16] Intel: *The Internet of Things (IoT) Starts with Intel Inside*. Electronic. http://www.intel.com/content/www/us/en/internet-of-things/overview.html?_ga=1.229616796.1463032877.1457448013. Version: 2016. – lasti visited August 17, 2017

[JKW15] Jones, Nick ; Kleynhans, Stephen ; Wallin, Leif-Olof: *Survey Analysis: The Internet of Things Is a Revolution Waiting to Happen*. Electronic. https://www.gartner.com/doc/2965320/survey-analysis-internet-things-revolution. Version: January 2015. – last visited on August 17, 2017

[JMSW02] Johnson, Robert ; Molnar, David ; Song, Dawn ; Wagner, David: Homomorphic signature schemes. In: *Topics in Cryptology—CT-RSA 2002*. Springer, 2002, S. 244–262

[Jue04] Juels, Ari: Minimalist cryptography for low-cost RFID tags. In: *International Conference on Security in Communication Networks* Springer, 2004, S. 149–164

[JWV13] Janic, Milena ; Wijbenga, Jan P. ; Veugen, Thijs: Transparency enhancing tools (TETs): an overview. In: *2013 Third Workshop on Socio-Technical Aspects in Security and Trust* IEEE, 2013, S. 18–25

[KBC97] Krawczyk, Hugo ; Bellare, Mihir ; Canetti, Ran: *RFC 2104: HMAC: Keyed-hashing for message authentication*. 1997

[KBG13] Korzun, Dmitry G. ; Balandin, Sergey I. ; Gurtov, Andrei V.: Deployment of Smart Spaces in Internet of Things: Overview of the design challenges. In: *Internet of Things, Smart Spaces, and Next Generation Networking*. Springer, 2013, S. 48–59

[KFJ01] Kagal, L. ; Finin, T. ; Joshi, A.: Trust-based security in pervasive computing environments. In: *Computer* 34 (2001), Dec, Nr. 12, S. 154–157. http://dx.doi.org/10.1109/2.970591. – DOI 10.1109/2.970591. – ISSN 0018–9162

[KKFS10] Kortuem, G. ; Kawsar, F. ; Fitton, D. ; Sundramoorthy, V.: Smart objects as building blocks for the Internet of things. In: *Internet Computing, IEEE* 14 (2010), Jan, Nr. 1, S. 44–51. http://dx.doi.org/10.1109/MIC.2009.143. – DOI 10.1109/MIC.2009.143. – ISSN 1089–7801

[KKG08] Kalloniatis, Christos ; Kavakli, Evangelia ; Gritzalis, Stefanos: Addressing privacy requirements in system design: the PriS method. In: *Requir. Eng.* 13 (2008), August, S. 241–255

[KKZK12] Khan, Raees ; Khan, Samee U. ; Zaheer, Rifaqat ; Khan, Sharifullah: Future Internet: the Internet of Things architecture, possible applications and key challenges. In: *Frontiers of Information Technology (FIT), 2012 10th International Conference on* IEEE, 2012, S. 257–260

[KLD11] Kraemer, Kenneth ; Linden, Greg ; Dedrick, Jason: Capturing value in Global Networks: Apple's iPad and iPhone. In: *University of California, Irvine, University of California, Berkeley, y Syracuse University, NY. http://pcic. merage. uci. edu/papers/2011/value_iPad_iPhone. pdf. Consultado el* 15 (2011)

[KM03] Koetter, Ralf ; Médard, Muriel: An algebraic approach to network coding. In: *IEEE/ACM Transactions on Networking (TON)* 11 (2003), Nr. 5, S. 782–795

[KM14] Kohnstamm, Jacob ; Madhub, Drudeisha: Mauritius Declaration on the Internet of Things. In: *36th International Conference of Data Protection and Privacy Comissioners*, 2014

[Koo14] Koops, Bert-Jaap: The trouble with European data protection law. In: *International Data Privacy Law* 4 (2014), Nr. 4, S. 250–261

[KR00] Krawczyk, Hugo ; Rabin, Tal: Chameleon hashing and signatures. In: *Proc. of NDSS* Citeseer, 2000, S. 143–154

[KS10] Koshizuka, Noboru ; Sakamura, Ken: Ubiquitous ID: standards for ubiquitous computing and the Internet of Things. In: *IEEE Pervasive Computing* (2010), Nr. 4, S. 98–101

[KSH+12] Kothmayr, Thomas ; Schmitt, Corinna ; Hu, Wen ; Brunig, Michael ; Carle, Georg: A DTLS based end-to-end security architecture for the Internet of Things with two-way authentication. In: *Local Computer Networks Workshops (LCN Workshops), 2012 IEEE 37th Conference on* IEEE, 2012, S. 956–963

[LKR04] Lu, Gang ; Krishnamachari, Bhaskar ; Raghavendra, Cauligi S.: Performance evaluation of the IEEE 802.15. 4 MAC for low-rate low-power wireless networks. In: *Performance, Computing, and Communications, 2004 IEEE International Conference on* IEEE, 2004, S. 701–706

[LLC15] LLC, Ponemon I.: 2015 Cost of Data Breach Study: Global Analysis / Ponemon Institute LLC. Version: 2015. https://securityintelligence.com/cost-of-a-data-breach-2015/. 2015. – Forschungsbericht

[LLZ+14] Li, Hongwei ; Lu, Rongxing ; Zhou, Liang ; Yang, Bo ; Shen, Xuemin: An efficient merkle-tree-based authentication scheme for smart grid. In: *Systems Journal, IEEE* 8 (2014), Nr. 2, S. 655–663

[Lor97] Lorand, Erric: *Appendix 1. Breakdown of costs for a pair of Nike shoes from an Indonesian plant.* Electronic. http://www-personal.umich.edu/~lormand/poli/nike/nike101-8.htm. Version: 1997

[LRSW99] Lysyanskaya, Anna ; Rivest, Ronald L. ; Sahai, Amit ; Wolf, Stefan: Pseudonym systems. In: *International Workshop on Selected Areas in Cryptography* Springer, 1999, S. 184–199

[LSHP06] Li, Mingyan ; Sampigethaya, Krishna ; Huang, Leping ; Poovendran, Radha: Swing & swap: user-centric approaches towards maximizing location privacy. In: *Proceedings of the 5th ACM workshop on Privacy in electronic society* ACM, 2006, S. 19–28

[Lyo77] Lyons, John: *Semantics (Vol. 1 & Vol. 2).* 1977

[May09a] Mayer, Christoph P.: Security and Privacy Challenges in the Internet of Things. In: *Workshops der Wissenschaftlichen Konferenz - Kommunikation in Verteilten Systemen* (2009), S. 12

[May09b] Mayer, Christoph P.: Security and privacy challenges in the internet of things. In: *Electronic Communications of the EASST* 17 (2009)

[MBG+08] McCoy, Damon ; Bauer, Kevin ; Grunwald, Dirk ; Kohno, Tadayoshi ; Sicker, Douglas: Shining light in dark places: Understanding the Tor network. In: *Privacy Enhancing Technologies* Springer, 2008, S. 63–76

[McC04] McCormick, Robert: Collaboration: The challenge of ICT. In: *International Journal of Technology and Design Education* 14 (2004), Nr. 2, S. 159–176

[McC09] McClusky, Mark: The Nike experiment: How the shoe giant unleashed the power of personal metrics. In: *Wired* 17 (2009), Nr. 07

[MCM09] McGraw, Gary ; Chess, Brian ; Migues, Sammy: *Building security in maturity model*. 2009

[Me16] Me, Diconnect: *Privacy policies are too complicated - We've simplified them*. Electronic. https://disconnect.me/icons. Version: 2016. – last accessed on August 17, 2017

[MEH01] Maier, Mark W. ; Emery, David ; Hilliard, Rich: Software architecture: introducing IEEE Standard 1471. In: *Computer* 34 (2001), Nr. 4, S. 107–109

[Mer79] Merkle, Ralph C.: Secrecy, authentication, and public key systems. (1979)

[Mer90] Merkle, Ralph C.: A certified digital signature. In: *Advances in Cryptology—CRYPTO'89 Proceedings* Springer, 1990, S. 218–238

[MFG+12] Manulis, Mark ; Fleischhacker, Nils ; Gunther, Felix ; Kiefer, Franziskus ; Poettering, Bertram: Group Signatures: Authentication with Privacy / Cryptographic Protocols Group Department of Computer Science, Technische Universitaet Darmstadt. Version: 2012. https://www.bsi.bund.de/SharedDocs/Downloads/EN/BSI/Publications/Studies/GruPA/GruPA.pdf;

jsessionid=F067F8B869D36803F1BDB69E44C20467.2_cid368?___blob=publicationFile. 2012. – Forschungsbericht

[MHJH15a] Moskowitz, R ; Heer, T ; Jokela, P ; Henderson, T: Host identity protocol version 2 (HIPv2). 2015. – Forschungsbericht

[MHJH15b] Moskowitz, R. ; Heer, T. ; Jokela, P. ; Henderson, T.: Host Identity Protocol Version 2 (HIPv2) / IETF. 2015. – IETF Draft

[Mic02] Microsytems, Sun ; Sun Microsystems, Inc. (Hrsg.): *Java Card 2.2 Virtual Machine Specification.* 901 San Antonio Road Palo Alto, CA 94303 USA: Sun Microsystems, Inc., June 2002. https://www.informatik.uni-augsburg.de/lehrstuehle/swt/se/teaching/fruehere_semester/ws0708/javacard/Dokumentation/JcvmSpec.pdf

[Mic16] Microsoft: *The Internet of your things - The Internet of Things (IoT) brings together devices, sensors, cloud, data and your imagination.* Electronic. https://dev.windows.com/en-us/iot. Version: 2016. – last visited on August 17, 2017

[MKHC07] Montenegro, Gabriel ; Kushalnagar, Nandakishore ; Hui, Jonathan ; Culler, David: Transmission of IPv6 packets over IEEE 802.15. 4 networks. 2007. – Forschungsbericht

[MKW08] Ma, Zhendong ; Kargl, Frank ; Weber, Michael: Pseudonym-on-demand: a new pseudonym refill strategy for vehicular communications. In: *Vehicular Technology Conference, 2008. VTC 2008-Fall. IEEE 68th* IEEE, 2008, S. 1–5

[MM92] Mattessich, Paul W. ; Monsey, Barbara R.: *Collaboration: what makes it work. A review of research literature on factors influencing successful collaboration.* ERIC, 1992

[MNJH08] Moskowitz, Robert ; Nikander, Pekka ; Jokela, Petri ; Henderson, Thomas: Host identity protocol. 2008. – Forschungsbericht

[MR02] Micali, Silvio ; Rivest, Ronald L.: Transitive signature schemes. In: *Topics in Cryptology—CT-RSA 2002.* Springer, 2002, S. 236–243

[MS01] Millen, Jonathan ; Shmatikov, Vitaly: Constraint Solving for Bounded-process Cryptographic Protocol Analysis. In: *Proceedings of the 8th ACM Conference on Computer and Communications*

Security. New York, NY, USA : ACM, 2001 (CCS '01). – ISBN 1–58113–385–5, 166–175

[MS08] Marwane, El K. ; Stein, Sebastian: Policy-Based Semantic Compliance Checking for Business Process Management. In: *Proceedings of the Workshops co-located with the Conference on Modellierung betrieblicher Informationssysteme (MobIS)* Bd. 420, CEUR-WS.org, 2008 (CEUR Workshop Proceedings), S. 178–192

[MS10] Medaglia, CarloMaria ; Serbanati, Alexandru: An Overview of Privacy and Security Issues in the Internet of Things. Version: 2010. dx.doi.org/10.1007/978-1-4419-1674-7_38. In: Giusto, Daniel (Hrsg.) ; Iera, Antonio (Hrsg.) ; Morabito, Giacomo (Hrsg.) ; Atzori, Luigi (Hrsg.): *The Internet of Things*. Springer New York, 2010. – ISBN 978–1–4419–1673–0, 389-395

[MSDL99] Mitchell, J ; Scedrov, A ; Durgin, N ; Lincoln, P: Undecidability of bounded security protocols. In: *Workshop on Formal Methods and Security Protocols*, 1999

[MSPC12] Miorandi, Daniele ; Sicari, Sabrina ; Pellegrini, Francesco D. ; Chlamtac, Imrich: Internet of things: Vision, applications and research challenges. In: *Ad Hoc Networks* 10 (2012), Nr. 7, 1497 - 1516. http://dx.doi.org/http://dx.doi.org/10.1016/j.adhoc.2012.02.016. – DOI http://dx.doi.org/10.1016/j.adhoc.2012.02.016. – ISSN 1570–8705

[MSW05] Molnar, David ; Soppera, Andrea ; Wagner, David: A scalable, delegatable pseudonym protocol enabling ownership transfer of RFID tags. In: *International Workshop on Selected Areas in Cryptography* Springer, 2005, S. 276–290

[MTF+] Moldovan, George ; Tragos, Elias Z. ; Fragkiadakis, Alexandros ; Pöhls, Henrich C. ; Calvo, Daniel: An IoT middleware for enhanced security and privacy: the RERUM approach.

[Mul07] Mulligan, Geoff: The 6LoWPAN architecture. In: *Proceedings of the 4th workshop on Embedded networked sensors* ACM, 2007, S. 78–82

[Mul08] Muller, Gerrit: A reference architecture primer. In: *Eindhoven Univ. of Techn., Eindhoven, White paper* (2008)

[Mus11] Musil, Steven: *Hackers claim to have stolen PSN credit card info*. Electronic. http://www.cnet.com/news/hackers-claim-to-have-stolen-psn-credit-card-info/. Version: April 2011

[MW04] Molnar, David ; Wagner, David: Privacy and security in library RFID: Issues, practices, and architectures. In: *Proceedings of the 11th ACM conference on Computer and communications security* ACM, 2004, S. 210–219

[MY15] Mathews, Anna W. ; Yadron, D: Health insurer anthem hit by hackers. In: *The Wall Street Journal* (2015)

[NAD+08] Noshadi, Hyduke ; Ahmadian, Shaun ; Dabiri, Foad ; Nahapetian, Ani ; Stathopoulus, Thanos ; Batalin, Maxim ; Kaiser, William ; Sarrafzadeh, Majid: Smart shoe for balance, fall risk assessment and applications in wireless health. In: *Microsoft eScience Workshop (December 2008)*, 2008

[Not15] Notario, N. et a.: PRIPARE: Integrating Privacy Best Practices into a Privacy Engineering Methodology. In: *Security and Privacy Workshops (SPW), 2015 IEEE* IEEE, 2015, S. 151–158

[NS78] Needham, Roger M. ; Schroeder, Michael D.: Using encryption for authentication in large networks of computers. In: *Communications of the ACM* 21 (1978), Nr. 12, S. 993–999

[NSFB15] Neisse, Ricardo ; Steri, Gary ; Fovino, Igor N. ; Baldini, Gianmarco: SecKit: A model-based security toolkit for the internet of things. In: *Computers & Security* 54 (2015), S. 60–76

[NY89] Naor, Moni ; Yung, Moti: Universal one-way hash functions and their cryptographic applications. In: *Proceedings of the twenty-first annual ACM symposium on Theory of computing* ACM, 1989, S. 33–43

[OAG+11] Oliveira, Leonardo B. ; Aranha, Diego F. ; Gouvêa, Conrado P. ; Scott, Michael ; Câmara, Danilo F. ; López, Julio ; Dahab, Ricardo: TinyPBC: Pairings for authenticated identity-based non-interactive key distribution in sensor networks. In: *Computer Communications* 34 (2011), Nr. 3, S. 485–493

[Oli14] Oliver, Dr I.: *Privacy Engineering: A Dataflow and Ontological Approach*. 1st. USA : CreateSpace Independent Publishing Platform, 2014

[OPW+12] Oppenheim, Chasey ; Patrick, Jackson ; Warren, Gus ; Toyens, Victor ; Goodale, Eason ; Mills, Codi ; Grannum, Errol: *Disconnect.me: Privacy Icon crowdsourcing effort*. Web Page. https://disconnect.me/icons. Version: 2012. – (accessed 2015/07/21)

[OSK+03] Ohkubo, Miyako ; Suzuki, Koutarou ; Kinoshita, Shingo u. a.: Cryptographic approach to "privacy-friendly" tags. In: *RFID privacy workshop* Bd. 82 Cambridge, USA, 2003

[Par02] Parliament, E: Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector, Off. In: *JL 201, 31.7. 2002, at 37.(Directive on Privacy and Electronic Communications)* (2002)

[Par14] Parliament, European: *Legislative resolution of 12 March 2014 on the proposal for a regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation)*. Web Page. http://dx.doi.org/P7_TA-PROV(2014)0212,A7-0402/2013. Version: March 2014. – (accessed 2015/07/14)

[PCTS05] Perrig, Adrian ; Canetti, Ran ; Tygar, J D. ; Song, Dawn: The TESLA broadcast authentication protocol. In: *RSA CryptoBytes* 5 (2005)

[Per98] Perkins, Charles E.: Mobile networking through mobile IP. In: *Internet Computing, IEEE* 2 (1998), Nr. 1, S. 58–69

[Per08] Perez, Juan C.: *Facebook's Beacon More Intrusive Than Previously Thought*. Electronic. http://www.pcworld.com/article/140182/article.html. Version: 2008

[PLB12] Pretschner, Alexander ; Lovat, Enrico ; Büchler, Matthias: Representation-independent data usage control. In: *Data Privacy Management and Autonomous Spontaneus Security*. Springer, 2012, S. 122–140

[PM11] Pearson, Siani ; Mont, Marco C.: Sticky policies: an approach for managing privacy across multiple parties. In: *Computer* (2011), Nr. 9, S. 60–68

[PPS+13] Pöhls, Henrich C. ; Peters, Stefan ; Samelin, Kai ; Posegga, Joachim ; Meer, Hermann de: Malleable signatures for resource constrained platforms. In: *Information Security Theory and Practice. Security of Mobile and Cyber-Physical Systems*. Springer, 2013, S. 18–33

[PRRJ03] Potlapally, Nachiketh R. ; Ravi, Srivaths ; Raghunathan, Anand ; Jha, Niraj K.: Analyzing the energy consumption of security protocols. In: *Proceedings of the 2003 international symposium on Low power electronics and design* ACM, 2003, S. 30–35

[PTRC07] Peffers, Ken ; Tuunanen, Tuure ; Rothenberger, Marcus A. ; Chatterjee, Samir: A Design Science Research Methodology for Information Systems Research. In: *Journal of Management Information Systems* 24 (2007), 45-77. http://dx.doi.org/10.2753-MIS0742-1222240302. – DOI 10.2753–MIS0742–1222240302

[Qui11] Quitmann, Kristina: *Eigentums- und Besitzschutz im deutschen und englischen Recht.: Rechtsvergleichende Analyse des Spannungsverhältnisses zwischen Eigentum und Besitz.*. Bd. 1. Duncker & Humblot; Auflage: 1, 14. Dezember 2011. – 384 S.

[QV04] Qu, Minghua ; Vanstone, Scott A.: *Implicit certificate scheme*. September 14 2004. – US Patent 6,792,530

[Rag11] Raggett, Dave: *Privacy Enhancing Browser Extensions*. Electronic. https://www.w3.org/2011/D1.2.3/#dashboard. Version: February 2011. – last visited on August 17, 2017

[Rei94] Reidenberg, Joel R.: Setting standards for fair information practice in the US private sector. In: *Iowa L. Rev.* 80 (1994), S. 497

[RER14a] RERUM: Deliverable 2.1 - Use-cases definition and threat analysis / RERUM Project. Version: 2014. https://bscw.ict-rerum.eu/pub/bscw.cgi/d14540/RERUM_deliverable_D2_1_rev1_1.pdf. 2014. – Forschungsbericht

[RER14b] RERUM: Deliverable 2.3 - System Architecture / RERUM Project. Version: 2014. https://bscw.ict-rerum.eu/pub/bscw.cgi/

d18321/RERUM%20deliverable%20D2_3.pdf. 2014. – Forschungsbericht. – last visited on August 17, 2017

[RER15] RERUM: Deliverable 2.5 - Final System Architecture / RERUM Project. Version: 2015. https://bscw.ict-rerum.eu/pub/bscw.cgi/d31979/RERUM%20deliverable%20D2_5.pdf. 2015. – Forschungsbericht

[Res14] Resarch, Certicom: Standards for Efficient Cryptography 4 (SEC4): Elliptic Curve Qu-Vanstone Implicit Certificate Scheme (ECQV) / Certicom Resarch. 2014 (Version (Draft) 1.2). – Working Draft. – Available at http://www.secg.org/sec4-1.0.pdf, last accessed on 2015/11/10.

[RG13] Rubinstein, Ira S. ; Good, Nathaniel: Privacy by design: A counterfactual analysis of Google and Facebook privacy incidents. In: *Berkeley Tech. LJ* 28 (2013), S. 1333

[RHCF05] Rayner, Manny ; Hockey, Beth A. ; Chatzichrisafis, Nikos ; Farrell, Kim: OMG unified modeling language specification. In: *Version 1.3,© 1999 Object Management Group, Inc* Citeseer, 2005

[Ris13] Rissanen, Erik: eXtensible Access Control Markup Language (XACML) Version 3.0 / OASIS. Version: January 2013. http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-os-en.html. 2013 (urn:oasis:names:tc:xacml:3.0:core:schema:wd-17). – Standard. – (accessed 2015/07/26)

[RNL11] Roman, R. ; Najera, P. ; Lopez, J.: Securing the Internet of Things. In: *Computer* 44 (2011), Sept, Nr. 9, S. 51–58. http://dx.doi.org/10.1109/MC.2011.291. – DOI 10.1109/MC.2011.291. – ISSN 0018–9162

[RPP11] Rohokale, Vandana M. ; Prasad, Neeli R. ; Prasad, Ramjee: A cooperative Internet of Things (IoT) for rural healthcare monitoring and control. In: *Wireless Communication, Vehicular Technology, Information Theory and Aerospace & Electronic Systems Technology (Wireless VITAE), 2011 2nd International Conference on* IEEE, 2011, S. 1–6

[RSM+01] Ryan, Peter ; Schneider, Steve A. ; M.H., Goldsmith ; G., Lowe ; A.W., Roscoe ; Ryan, P.Y.A (Hrsg.) ; Schneider, S.A. (Hrsg.):

The modelling and analysis of security protocols: the csp approach. Bd. 1. 1. Addison-Wesley Professional, 2001. – ISBN 0201674718

[RW12] Rozanski, Nick ; Woods, Eóin: *Software systems architecture: working with stakeholders using viewpoints and perspectives*. Bd. 2nd revised edition. Addison-Wesley, 2012. – 34 S.

[RZL13] Roman, Rodrigo ; Zhou, Jianying ; Lopez, Javier: On the features and challenges of security and privacy in distributed internet of things. In: *Computer Networks* 57 (2013), Nr. 10, 2266 - 2279. http://dx.doi.org/http://dx.doi.org/10.1016/j.comnet.2012.12.018. – DOI http://dx.doi.org/10.1016/j.comnet.2012.12.018. – ISSN 1389–1286. – Towards a Science of Cyber Security Security and Identity Architecture for the Future Internet

[San15] Sangkeun, YOO: ISO/IEC JTC 1/WG 10: Working Group on Internet of Things / ISO/IEC. Version: 2015. http://iot-week.eu/wp-content/uploads/2015/06/07-JTC-1-WG-10-Introduction.pdf. 2015 (ISO/IEC JTC 1/WG 10). – Forschungsbericht. – last visited August 17, 2017.

[SBJ+14] Sakimura, Natsuhiko ; Bradley, J ; Jones, M ; Medeiros, B de ; Mortimore, C: Openid connect core 1.0. In: *The OpenID Foundation* (2014), S. S3

[SBZ02] Steinfeld, Ron ; Bull, Laurence ; Zheng, Yuliang: Content extraction signatures. In: *Information Security and Cryptology—ICISC 2001* (2002), S. 163–205

[SC09] Spiekermann, Sarah ; Cranor, Lorrie F.: Engineering privacy. In: *Software Engineering, IEEE Transactions on* 35 (2009), Nr. 1, S. 67–82

[SC15] Suppan, Santiago ; Cuellar, Jorge: Datenschutzbewahrende Policies für die Kommunikation im Internet der Dinge. In: *Prior Art Journal* 4 (2015). http://dx.doi.org/10.4421/PAPDEOTT004112. – DOI 10.4421/PAPDEOTT004112

[SC16] Suppan, Santiago ; Cuéllar, Jorge ; Grace, Jennifer (Hrsg.): *Privacy Enhancing Technologies for the Internet of Things*. Bd. 1. London, UK. : The Institution of Engineering and Technology, 2016. – 25 S.

[Sch78] Schauer, Frederick: Fear, risk and the first amendment: Unraveling the chilling effect. In: *BUL Rev.* 58 (1978), S. 685

[SG09] Sarma, Amardeo C. ; Girão, João: Identities in the future internet of things. In: *Wireless personal communications* 49 (2009), Nr. 3, S. 353–363

[SGFW10] Sundmaeker, Harald ; Guillemin, Patrick ; Friess, Peter ; Woelfflé, Sylvie: *Vision and challenges for realising the Internet of Things*. EUR-OP, 2010 http://bookshop.europa.eu/en/vision-and-challenges-for-realising-the-internet-of-things-pbKK3110323/downloads/KK-31-10-323-EN-C/KK3110323ENC_002.pdf;pgid=y8dIS7GUWMdSR0EAlMEUUsWb0000HS_tN3Oi;sid=qJ_daQe-6Hvdelce-uhDzmWb5fzDG0rdtu0=?FileName=KK3110323ENC_002.pdf&SKU=KK3110323ENC_PDF&CatalogueNumber=KK-31-10-323-EN-C

[SHBF12] Shelby, Z ; Hartke, K ; Bormann, C ; Frank, B: Constrained Application Protocol (CoAP), draft-ietf-core-coap-13. In: *Orlando: The Internet Engineering Task Force–IETF, Dec* (2012)

[Sir14] Siriwardena, Prabath: OAuth 2.0 Profiles. In: *Advanced API Security*. Springer, 2014, S. 143–153

[SKH+15] Soldatos, John ; Kefalakis, Nikos ; Hauswirth, Manfred ; Serrano, Martin ; Calbimonte, Jean-Paul ; Riahi, Mehdi ; Aberer, Karl ; Jayaraman, Prem P. ; Zaslavsky, Arkady ; Žarko, Ivana P. u. a.: Openiot: Open source internet-of-things in the cloud. In: *Interoperability and Open-Source Solutions for the Internet of Things*. Springer, 2015, S. 13–25

[SKP+11] Schaffers, Hans ; Komninos, Nicos ; Pallot, Marc ; Trousse, Brigitte ; Nilsson, Michael ; Oliveira, Alvaro: Smart Cities and the Future Internet: Towards Cooperation Frameworks for Open Innovation. In: *Future internet assembly* 6656 (2011), Nr. 31, S. 431–446

[SMH+10] Serrano, Martin ; Meer, Sven van d. ; Holum, V ; Murphy, J ; Strassner, John: Federation, a matter of autonomic management in the Future Internet. In: *Network Operations and Management Symposium (NOMS), 2010 IEEE* IEEE, 2010, S. 845–849

[SNCR03] Srinivasan, Vikram ; Nuggehalli, Pavan ; Chiasserini, Carla F. ; Rao, Rohini R.: Cooperation in wireless ad hoc networks. In: *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies* Bd. 2 IEEE, 2003, S. 808–817

[SNS88] Steiner, Jennifer G. ; Neuman, B C. ; Schiller, Jeffrey I.: Kerberos: An Authentication Service for Open Network Systems. In: *Usenix Winter*, 1988, S. 191–202

[Sol06] Solove, Daniel J.: A taxonomy of privacy. In: *University of Pennsylvania law review* (2006), S. 477–564

[Sol07] Solove, Daniel J.: 'I've got nothing to hide' and other misunderstandings of privacy. In: *San Diego law review* 44 (2007), S. 745

[Sol12] Solove, Daniel J.: Introduction: Privacy self-management and the consent dilemma. In: *Harv. L. Rev.* 126 (2012), S. 1880

[Sol15] Solove, Daniel J.: 10 Implications of the New EU General Data Protection Regulation (GDPR). In: *Tech Privacy - Privavy + Security Blog* (2015). https://www.teachprivacy.com/new-eu-data-protection-regulation-gdpr/

[Son15] Sony Corporation: *Consolidated Financial Results Forecast for the Third Quarter*. Electronic. http://www.sony.net/SonyInfo/IR/library/fr/150204_sony.pdf. Version: February 2015

[SPF08] Scheuer, Florian ; Posse, K ; Federrath, Hannes: Preventing profile generation in vehicular networks. In: *Networking and Communications, 2008. WIMOB'08. IEEE International Conference on Wireless and Mobile Computing,* IEEE, 2008, S. 520–525

[SS11] Segura, Alexander S. ; SAG, Norbert V.: Internet of Things Architecture IoT-A Project Deliverable D6. 2–Updated Requirements List. (2011)

[Sta15] Statista: *Number of free mobile app downloads worldwide from 2012 to 2017*. http://www.statista.com/statistics/241587/number-of-free-mobile-app-downloads-worldwide/. Version: 2015

[STBW02] Schäfer, Ralf-Peter ; Thiessenhusen, Kai-Uwe ; Brockfeld, Elmar ; Wagner, Peter: A traffic information system by means of real-time floating-car data. In: *ITS World Congress 2002*, 2002

[STC+13] Schulzrinne, H ; Tschofenig, H ; Cuellar, J ; Polk, J ; Morris, J ; Thomson, M: Geolocation policy: A document format for expressing privacy preferences for location information. 2013. – Forschungsbericht

[SU05] Strategy, ITU ; Unit, Policy: Executive Summary - The Internet of Things / ITU Strategy and Policy Unit. Version: 2005. http://www.internet-of-things.eu/resources/documents/iotsummary.pdf/at_download/file. 2005. – Forschungsbericht

[SWC+15] Suppan, Santiago ; Weber, Ricarda ; Cuellar, Jorge ; Staudemeye, Ralf C. ; Lopez, Dario R. ; Wojcik, Marcin ; Poehls, Henrich C. ; Bauer, Johannes ; Petschkuh, Benedikt ; Fragkiadaki, Alexandros ; Tragos, Elias: Privacy enhancing techniques in the Smart City applications / RERUM Project. Version: September 2015. https://bscw.ict-rerum.eu/pub/bscw.cgi/d31975/RERUM%20deliverable%20D3_2.pdf. 2015. – Forschungsbericht

[TA08] Thaler, Dave ; Aboba, Bernard: What makes for a successful protocol? 2008. – Forschungsbericht

[Tec12] Tech, Tactical: *Me and my shadow - take control of your data.* Electronic. https://myshadow.org/. Version: 2012. – last accessed on August 17, 2017

[TEGIE13] Things Expert Group (IoT-EG), Internet of: *IoT Privacy, Data Protection, Information Security.* Electronic. http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=1753. Version: 2013

[Tha14] Thatmann, Dirk: Distributed authorization in complex multi entity-driven API ecosystems. In: *Signal Processing and Communication Systems (ICSPCS), 2014 8th International Conference on* IEEE, 2014, S. 1–9

[TPS+15] Tragos, Elias Z. ; Pöhls, Henrich C. ; Staudemeyer, Ralf ; Slamanig, Daniel ; Kapovits, Adam ; Suppan, Santiago ; Fragkiadakis, Alexandros ; Baldini, Gianmarco ; Neisse, Ricardo ;

Langendörfer, Peter: *Securing the Internet of Things - Security and Privacy in a Hyperconnected World*. Bd. 43. River Publishers, 2015. – ISBN 978–87–93237–99–5

[TPT06] Titkov, Leonid ; Poslad, Stefan ; Tan, Juan J.: An integrated approach to user-centered privacy for mobile information services. In: *Applied Artificial Intelligence* 20 (2006), Nr. 2-4, S. 159–178

[TSF12] Tomandl, Andreas ; Scheuer, Florian ; Federrath, Hannes: Simulation-based evaluation of techniques for privacy protection in VANETs. In: *Wireless and Mobile Computing, Networking and Communications (WiMob), 2012 IEEE 8th International Conference on* IEEE, 2012, S. 165–172

[Uni14] United Nations: *World Urbanization Prospects*. Electronic. http://esa.un.org/unpd/wup/highlights/wup2014-highlights.pdf. Version: 2014. – lasti visited August 17, 2017

[Uri09] Urien, P.: *Internet Draft: HIP for IoT*. 2009

[VF10] Vermesan, Ovidiu ; Friess, Peter: European research cluster on the internet of things. In: *Internet of Things-Global Technological and Societal Trends From Smart Environments and Spaces to Green ICT* (2010), S. 1

[VFG+11] Vermesan, Ovidiu ; Friess, Peter ; Guillemin, Patrick ; Gusmeroli, Sergio ; Sundmaeker, Harald ; Bassi, Alessandro ; Jubert, Ignacio S. ; Mazura, Margaretha ; Harrison, Mark ; Eisenhauer, Markus ; Doody, Pat: Internet of Things Strategic Research Roadmap / European Research Cluster on the Internet of Things. Version: 2011. http://internet-of-things-research.eu/pdf/IoT_Cluster_Strategic_Research_Agenda_2011.pdf. 2011. – Forschungsbericht

[VKC15] Varma, Atul ; Kovacs, Gary ; Carr, Emily: *Lightbeam for Firefox*. Web Page. https://www.mozilla.org/de/lightbeam/. Version: July 2015. – (accessed 2015/07/16)

[VMZS+13] Vidal Meca, F. ; Ziegeldorf, J.H. ; Sanchez, P.M. ; Morchon, O.G. ; Kumar, S.S. ; Keoh, S.L.: HIP Security Architecture for the IP-Based Internet of Things. In: *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, 2013, S. 1331–1336

[WBC+09] Welbourne, E. ; Battle, L. ; Cole, G. ; Gould, K. ; Rector, K. ; Raymer, S. ; Balazinska, M. ; Borriello, G.: Building the Internet of Things Using RFID: The RFID Ecosystem Experience. In: *Internet Computing, IEEE* 13 (2009), May, Nr. 3, S. 48–55. http://dx.doi.org/10.1109/MIC.2009.52. – DOI 10.1109/MIC.2009.52. – ISSN 1089–7801

[Web09] Weber, Rolf H.: Internet of things–Need for a new legal environment? In: *Computer law & security review* 25 (2009), Nr. 6, S. 522–527

[Web10a] Weber, Rolf H.: Internet of Things – New security and privacy challenges. In: *Computer Law & Security Review* 26 (2010), Nr. 1, 23 - 30. http://dx.doi.org/http://dx.doi.org/10.1016/j.clsr.2009.11.008. – DOI http://dx.doi.org/10.1016/j.clsr.2009.11.008. – ISSN 0267–3649

[Web10b] Weber, Rolf H.: Internet of Things–New security and privacy challenges. In: *Computer Law & Security Review* 26 (2010), Nr. 1, S. 23–30

[Wee16] Wee, Alex: Google Brillio OS-another OS dedicated to Internet of Things. In: *nexus* (2016)

[Weg15] Wegener, Dieter: *Industrie 4.0-die Zukunft der digitalen Fabrik*. Technische Universität Dresden, Professur für Verarbeitungsmaschinen und Verarbeitungstechnik, 2015

[WGE+05] Wander, A. S. ; Gura, N. ; Eberle, H. ; Gupta, V. ; Shantz, S. C.: Energy analysis of public-key cryptography for wireless sensor networks. In: *Third IEEE International Conference on Pervasive Computing and Communications*, 2005, S. 324–328

[Wil14] Willis, Lauren E.: Why Not Privacy by Default. In: *Berkeley Tech. LJ* 29 (2014), S. 61

[WJ12] Wang, Qihua ; Jin, Hongxia: An analytical solution for consent management in patient privacy preservation. In: *Proceedings of the 2nd ACM SIGHIT International Health Informatics Symposium* ACM, 2012, S. 573–582

[WOT07] WOT: *Web of Trust (WOT) – Crowdsourced web safety*. Electronic. https://www.mywot.com/. Version: 2007. – last accessed on August 17, 2017

[WPJ+05] Walker, Jan ; Pan, Eric ; Johnston, Douglas ; Adler-Milstein, Julia ; Bates, David W. ; Middleton, Blackford: The value of health care information exchange and interoperability. In: *HEALTH AFFAIRS-MILLWOOD VA THEN BETHESDA MA-* 24 (2005), S. W5

[WS10] Wästlund, E. ; S.F., Hübner: End user transparency tools: UI prototypes / KAU. 2010 (4.2.2). – Deliverable

[WSJ15] Wuyts, Kim ; Scandariato, Riccardo ; Joosen, Wouter: *LINDDUN Privacy Threat Modeling*. Web Page. https://distrinet.cs.kuleuven.be/software/linddun/. Version: July 2015. – (accessed 2015/07/14), KU Leuven, Belgium

[WSRE04] Weis, Stephen A. ; Sarma, Sanjay E. ; Rivest, Ronald L. ; Engels, Daniel W.: Security and privacy aspects of low-cost radio frequency identification systems. In: *Security in pervasive computing*. Springer, 2004, S. 201–212

[WW02] Webster, Jane ; Watson, Richard T.: Analyzing the past to prepare for the future: Writing a literature review. In: *Management Information Systems Quarterly* 26 (2002), Nr. 2, S. 3

[YDHP07] Yang, Hung-chih ; Dasdan, Ali ; Hsiao, Ruey-Lung ; Parker, D S.: Map-reduce-merge: simplified relational data processing on large clusters. In: *Proceedings of the 2007 ACM SIGMOD international conference on Management of data* ACM, 2007, S. 1029–1040

[YKKS13] Yashiro, Takeshi ; Kobayashi, Shinsuke ; Koshizuka, Noboru ; Sakamura, Ken: An internet of things (iot) architecture for embedded appliances. In: *Humanitarian Technology Conference (R10-HTC), 2013 IEEE Region 10* IEEE, 2013, S. 314–319

[YZV14] Yan, Zheng ; Zhang, Peng ; Vasilakos, Athanasios V.: A survey on trust management for Internet of Things. In: *Journal of Network and Computer Applications* 42 (2014), 120 - 134. http://dx.doi.org/https://doi.org/10.1016/j.jnca.2014.01.014. – DOI https://doi.org/10.1016/j.jnca.2014.01.014. – ISSN 1084–8045

[ZAM14] Zimmermann, Christian ; Accorsi, Rafael ; Müller, Günter: Privacy dashboards: reconciling data-driven business models and

privacy. In: *Availability, Reliability and Security (ARES), 2014 Ninth International Conference on* IEEE, 2014, S. 152–157

[ZGW05] Zhang, Daqing ; Gu, Tao ; Wang, Xiaohang: Enabling context-aware smart home with semantic web technologies. In: *International Journal of Human-friendly Welfare Robotic Systems* 6 (2005), Nr. 4, S. 12–20

[ZLL+08] Zhang, C. ; Lin, X. ; Lu, R. ; Ho, P. H. ; Shen, X.: An Efficient Message Authentication Scheme for Vehicular Communications. In: *IEEE Transactions on Vehicular Technology* 57 (2008), Nov, Nr. 6, S. 3357–3368. http://dx.doi.org/10.1109/TVT.2008.928581. – DOI 10.1109/TVT.2008.928581. – ISSN 0018–9545

[ZT06] Zhu, Larry ; Tung, Brian: Public key cryptography for initial authentication in Kerberos (PKINIT). (2006)

[Zuc07] Zuckerberg, Mark: *Thoughts on Beacon*. Electronic. https://www.facebook.com/notes/facebook/thoughts-on-beacon/7584397130. Version: December 2007